

PATENT ABSTRACTS OF JAPAN

(11)Publication number : **11-134085**

(43)Date of publication of application : **21.05.1999**

(51)Int.Cl.

G06F 3/00

G06F 17/60

(21)Application number : **10-232231**

(71)Applicant : **LANDIS & STAefa INC**

(22)Date of filing : **14.07.1998**

(72)Inventor : **HAN JAMES K
RAMIREZ PETE**

(30)Priority

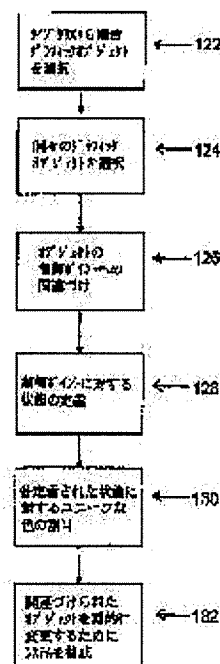
Priority number : **97 895801** Priority date : **17.07.1997** Priority country : **US**

(54) METHOD AND APPARATUS FOR MONITORING AND CONTROLLING REAL-TIME INFORMATION IN A BUILDING AUTOMATION SYSTEM

(57)Abstract:

PROBLEM TO BE SOLVED: To enable addition, change or deletion of a relation or a link between a graphic object and a control point by allocating state characteristics respectively corresponding to defined states, and dynamically displaying the state of the control point while using alarm characteristics.

SOLUTION: One composite graphic is selected from the library of stored compound graphics (block 122). One of independent selectable objects is selected (block 124). The selected objects are related (block 126). The state corresponding to the associated control point is instructed (block 128). A state color to be used for the control point to display the respective defined states is selected (block 130). A system guards against the dynamic change of the associated object state caused by refreshing the object displayed according to specified state characteristics (block 132).



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平11-134085

(43) 公開日 平成11年(1999) 5月21日

(51) Int.Cl.⁶G 0 6 F 3/00
17/60

識別記号

6 5 2

F I

G 0 6 F 3/00
15/216 5 2 C
K

審査請求 未請求 請求項の数17 O L 外国語出願 (全 52 頁)

(21) 出願番号 特願平10-232231

(22) 出願日 平成10年(1998) 7月14日

(31) 優先権主張番号 08/895801

(32) 優先日 1997年7月17日

(33) 優先権主張国 米国 (U S)

(71) 出願人 591268586

ランディス アンド スティーファ イン
コーポレイティドアメリカ合衆国、イリノイ州 60015、パ
ッファローグローブ、ディアフィールドパ
ークウェイ 1000

(72) 発明者 ジェームズ、ケー、ハン

アメリカ合衆国、イリノイ州60540、ナバ
ビル、サンクチュアリーレーン 932

(72) 発明者 ベテ、ラミレズ

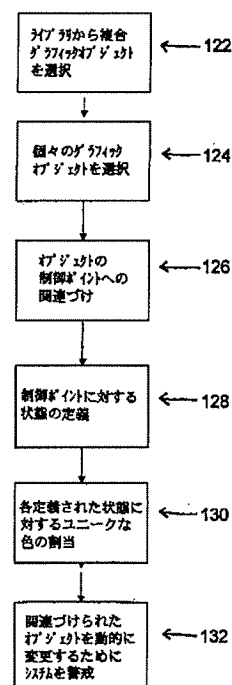
アメリカ合衆国、イリノイ州60137、グレン
エリン、スタプルフォードドライブ
148

(74) 代理人 弁理士 木村 高久

(54) 【発明の名称】 ビル自動システム内のリアルタイム情報をモニタし、制御する方法およびその装置

(57) 【要約】

この発明のビル自動制御システムは、少なくとも1つの周辺制御装置から処理情報（制御ポイント）をモニタし、制御する管理制御ステーションを有する。管理制御ステーションは、動的リンクライブラリ（DLL）を含むデータを格納するメモリが提供され、DLLは、独立して選択可能な複数のグラフィックオブジェクトを表示する第1のデータと、選択されたグラフィックオブジェクトと制御ポイントとの間の関係を特定するリンクデータとを有する。管理制御ステーションは、格納された静的テーブルデータ、リンクデータ、周辺制御装置からの実行時間値を用いて関連づけられた制御ポイントに対するリアルタイム処理状態情報を動的に表示する。



【特許請求の範囲】

【請求項1】 ビル自動システムでの動的処理情報をモニタし、制御するモジュラグラフィカル表示方法において、

複数の独立したグラフィックオブジェクトの中から表示されたグラフィックオブジェクトを選択し、

前記選択されたオブジェクトを制御ポイントにリンクし、

前記制御ポイントに対する通常状態および少なくとも1つのアラーム状態を定義し、

前記定義された状態のそれぞれに対応する状態特性を割り当て、

前記アラーム特性を用いて前記制御ポイントの状態を動的に表示することを特徴とする。

【請求項2】 前記状態特性は、状態カラーテーブルを用いて選択されることを特徴とする請求項1に記載のモジュラグラフィカル表示方法。

【請求項3】 前記状態特性は、表示されたテキストおよび前記グラフィックオブジェクトをブリンクすることを特徴とする請求項2に記載のモジュラグラフィカル表示方法。

【請求項4】 前記グラフィカルオブジェクトは、静的背景図でオーバーレイされることを特徴とする請求項1に記載のモジュラグラフィカル表示方法。

【請求項5】 各前記独立したグラフィックオブジェクトは、OCX制御であることを特徴とする請求項1に記載のモジュラグラフィカル表示方法。

【請求項6】 ビル自動システムでの動的処理情報をモニタし、制御するモジュラグラフィカル表示方法において、

情報ブロック制御、アナログバー制御、およびゲージ針制御からなるグループの中から表示されたグラフィックオブジェクトを選択し、該選択されたグラフィックオブジェクトは少なくとも1つの静的サブオブジェクトと少なくとも1つの動的サブオブジェクトとを有し、

前記選択されたグラフィックオブジェクトに対するスタイル特性を特定し、

前記選択されたオブジェクトを制御ポイントにリンクし、

前記制御ポイントに対する通常状態および少なくとも1つのアラーム状態を定義し、

各前記定義された状態に状態特性を割り当て、

前記状態特性に従った前記動的サブオブジェクトをリフレッシュして前記制御ポイントの実行時間値と状態情報とを動的に表示することを特徴とする。

【請求項7】 前記グラフィックオブジェクトはアナログバーであり、前記リンクされたポイントを命令するキャレットを有し、前記キャレットは動的サブオブジェクトであることを特徴とする請求項6に記載のモジュラグラフィカル表示方法。

【請求項8】 ユーザプロファイルが、ユーザが前記リンクされたポイントを命令したことを示した場合のみ、前記キャレットを移動可能とすることを特徴とする請求項7に記載のモジュラグラフィカル表示方法。

【請求項9】 前記アナログバーは、バー方向、サイズ、スケール、およびアラーム限界を含む、ユーザが調整可能な属性を有することを特徴とする請求項7に記載のモジュラグラフィカル表示方法。

【請求項10】 前記グラフィックオブジェクトは、前記ポイントの状態を命令するポイントコマンドに操縦されることができることを特徴とする請求項6に記載のモジュラグラフィカル表示方法。

【請求項11】 少なくとも1つの周辺制御装置と、CPU、データを格納するメモリ手段、前記周辺制御装置にデータを送信し該周辺制御装置からデータを受信する通信手段、グラフィカル情報を表示する表示手段、および入力装置を有し、前記周辺制御装置から少なくとも1つの制御ポイントに対するリアルタイム処理状態情報を受信する管理制御ステーションと、

選択された前記1つの制御ポイントに対する前記リアルタイム処理状態情報をグラフィカルに表示する少なくとも1つの制御オブジェクトとを有し、前記メモリ手段は、各前記少なくとも1つの制御オブジェクトに対する動的リンクライブラリ(DLL)を格納し、

前記DLLは、

1以上のOLEオブジェクトとして複数のサービスのそれぞれを定義するものであって、各前記OLEオブジェクトは1以上のインターフェースをサポートし、各前記インターフェースは幾つかのメソッドを有し、各メソッドは前記OLEオブジェクトのインターフェースを起動することによってのみ呼び出すことができる第1のデータと、

少なくとも1つの独立した選択可能なグラフィカルオブジェクトを表示する第2のデータと、

前記グラフィックオブジェクトと前記制御ポイントとの間の関係を特定するリンクデータと、

前記制御ポイントに対する状態の変更をグラフィカルに表示する状態情報のユーザが選択可能な変更を含む状態テーブルと、

表示されたフィールド、位置、およびサイズのうちの少なくとも1つを含む、オプションでユーザが選択した選択物を特定するスタイルデータとを具備し、

前記制御ステーションは、前記状態テーブルデータ、前記リンクデータ、および前記第1のデータを用いて前記制御ポイントに対する前記受信したリアルタイム処理状態情報を動的に表示することを特徴とするビル自動制御システム。

【請求項12】 各前記グラフィックオブジェクトは、静的サブオブジェクトおよび動的サブオブジェクトのグループから選択されたグラフィックオブジェクトからな

り、前記動的サブオブジェクトは前記静的サブオブジェクトから独立してリフレッシュされ、前記状態テーブルデータは、前記動的サブオブジェクトが仮定された各状態に対するデータを含むことを特徴とする請求項11に記載のビル自動制御システム。

【請求項13】 前記メモリ内に格納された静的背景図を表示するデータをさらに具備し、前記表示手段は、前記静的背景図上にオーバーレイされた少なくとも1つのグラフィックオブジェクトを表示し、前記グラフィックオブジェクトは前記静的背景図から独立してアドレスされ、リフレッシュされることを特徴とする請求項11に記載のビル自動制御システム。

【請求項14】 関連づけられたグラフィックオブジェクトを特定するハイパーリンクデータをさらに具備し、前記表示手段は、前記静的背景図上にオーバーレイされた少なくとも1つのグラフィックオブジェクトを表示し、前記グラフィックオブジェクトは前記静的背景図から独立してアドレスされ、リフレッシュされることを特徴とする請求項11に記載のビル自動制御システム。

【請求項15】 前記制御オブジェクトはアナログ制御であり、前記DLLは、前記アナログ制御の方向、アラームマーク、軸ラベル、ポイント名、および前記制御ポイントに対する表示された範囲の値を特定する高低チック数、を特定するアナログバースタイルデータさらに具備したことを特徴とする請求項11に記載のビル自動制御システム。

【請求項16】 前記制御オブジェクトは針ゲージ制御であり、前記DLLは、前記針ゲージ制御の方向、サイズ、および位置を特定するアナログバースタイルデータをさらに具備することを特徴とする請求項11に記載のビル自動制御システム。

【請求項17】 前記DLLは、複数の針スタイルの中から選択された針スタイルを特定するデータをさらに具備したことを特徴とする請求項16に記載のビル自動制御システム。

【発明の詳細な説明】

【0001】この発明の背景

この発明は、ビル自動システム内のリアルタイム情報をモニタし、制御する方法およびその装置に関する。特に、この発明は、動的にリアルタイム情報を表示する柔軟なモジュール装置を開示する。この発明はさらに、コードの一つの部分によってコードの他の部分をアクセスすることができ、これによってコードの再利用および維持の容易を促進できるひとつのアプローチを開示する。

【0002】グラフィカル・ディスプレイをモニタし、制御する従来のシステムは、柔軟性がなく、ユーザ・インターフェースの符号化での特定の制御ポイントで融通がきかずにリンクされあるいは関連づけられたグラフィ

カル制御あるいはイメージが用いられている。この内容では、制御ポイントは、モニタされ、あるいは命令される周辺制御装置のようなセンサや装置に接続される物理ポイントである。このリンクあるいは関係はインターフェースの中にコード化されていたので、従来のシステムは柔軟性がなく、追加の制御ポイントのその後の付加には、大きなプログラム修正が要求される。この柔軟性のなさは、ビル自動システムのアップグレードのコストに加わるという重大な欠点である。

【0003】従来のグラフィカル・ディスプレイ・システムに関連するさらなる欠点は、これらのグラフィカル制御あるいはイメージに関連する種々の属性が固定であることである。このため、制御、フォント、スケール等のサイズの変更は、基礎をなすプログラムの修正と再コンパイルのみによって達成することができる。

【0004】従来のグラフィカル・ディスプレイ・システムに関するさらなる欠点は、これらのシステムが他のソフトウェア、アプリケーション、ライブラリの一部によって提供されてこのソフトウェアサービスをアクセスすることに関する。従来のグラフィカル・モニタ・アプリケーションはライブラリにリンクされ、その後ライブラリの関数を呼び出してライブラリのサービスをアクセスする。あるいは、グラフィカル・ディスプレイ・アプリケーションは、完全に分離されたプロセスとして動作する他のアプリケーションによって提供されるサービスを使用する。このシナリオでは、2つのローカルなプロセスは、相互プロセス通信機構を用いて通信され、この相互プロセス通信機構は2つのアプリケーションのプロトコル（一方のアプリケーションに対して要求を特定し、他方のアプリケーションに対して適切に応答させるメッセージのセット）を要求する。他のシナリオでは、グラフィカル・ディスプレイ・アプリケーションはオペレーティングシステムによって提供されるサービスを使用する、すなわち、アプリケーションは、オペレーティングシステムによって処理される一連のシステムコールを作成する。

【0005】サービスを分割する標準アプローチがないので、従来のグラフィカル・モニタ・アプリケーションは、他のアプリケーションによって提供されるサービスをアクセスするための、数多くの異なったアプローチを含んでいる。さらに、すさまじいアプローチの使用は、コード部分の再利用を禁止し、ソフトウェアの維持を困難にする。このようにして、単一のアプローチは、一つのコード部分に、他のサービスをアクセスするのに必要とされる。

【0006】発明の要旨

従って、上述した問題に対応し、この発明の目的は、隠れたプログラミングお修正および／または再コンパイルの要求なしに、ユーザが、グラフィック・オブジェクトと制御ポイントとの間の関係あるいはリンクを付加、変

更、あるいは削除することができる柔軟性のあるモジュラー制御のセットをもつ、改良されたビル自動制御システムを提供することである。

【0007】この発明の他の目的は、隠れたプログラミングおよび／または再コンパイルなしに、ユーザが、グラフィック・オブジェクトに関する属性を修正することができるシステムを提供することである。

【0008】この発明の他の目的は、ソフトウェアの一部がそのサービスをソフトウェアの他の部分に供給し、これによりコード部分の再利用、ソフトウェアの開発と維持の容易を促進する単一のアプローチを含むグラフィカル・モニタ・アプリケーションを有したシステムを提供することである。

【0009】この発明のこれらおよび他の目的は、添付した図面を参照して、次に示すこの発明の詳細な説明から明らかになるだろう。

【0010】好ましい実施例の詳細な説明

端的に言えば、上述した目的は、ビル自動システムでリアルタイム情報をモニタするこのモジュラ・グラフィカル・ディスプレイ方法によって満たされ、超えられる。この発明の方法によって、ユーザは、複数の独立したオブジェクトの中から表示されたグラフィックオブジェクトを選択し、この選択されたオブジェクトを制御ポイントにリンクする。次に、ユーザは、関連する制御ポイントに対する通常状態と少なくとも1つのアラーム状態とを定義し、この定義された状態のそれぞれに対応する状態特性を割り当てる。最後に、制御ポイントの状態は、アラーム特性を用いた監視制御ステーション上で動的に表示される。

【0011】上述した目的は、ビル自動システムでリアルタイム情報をモニタし、制御するこの発明の装置によって満たされ、超えられる。一般的に、好ましい実施例では、この発明は、少なくとも1つの周辺の制御装置から受信した処理情報（制御ポイント）をモニタし、制御する監視制御ステーションを有するビル自動制御システムである。監視制御ステーションは、好ましくは、CPU、周辺制御装置からのデータおよび周辺制御装置へのデータを送受信する通信装置、グラフィカル情報を表示するディスプレイ、および入力装置を有する。監視制御ステーションは、通信装置を介して、周辺制御装置から、選択された制御ポイントのリアルタイム処理状態情報を受信する。さらに、リアルタイム処理状態情報は、制御オブジェクトを用いて動的に表示される。監視制御ステーションは、各制御オブジェクトに対する動的リンクライブラリ（DLL）を含むデータを格納するメモリが提供される。各制御オブジェクトのDLLは、複数の各サービスを1以上のオブジェクト指向でリンクされ、はめ込まれた（OLE）オブジェクトとして定義した第1データを有し、各OLEオブジェクトは1以上のインターフェースをサポートする。次に、各インターフェー

スは、OLEオブジェクトのインターフェースを起動することによってのみ呼び出すことができる幾つかのメソッドを有する。DLLは、さらに独立で選択可能な少なくとも1つのグラフィカルオブジェクトを表示する第2データを有し、選択されたグラフィックオブジェクトと制御ポイントの間の関係を特定するデータをリンクする。監視制御ステーションは、格納された状態テーブルデータ、リンクデータ、および周辺制御装置からの実行時間値を用いて、関連する制御ポイントのリアルタイム処理状態情報を動的に表示する。

【0012】この発明のグラフィカル・アプリケーション・デザイン・アーキテクチャによれば、ソフトウェアの各部分は、1以上のOLEオブジェクトとしてのサービスとして実装し、このOLEは、マイクロソフトのオブジェクト指向でリンクされ、はめ込まれた言語（OLE）であり、マイクロソフトの登録商標である。各OLEオブジェクトは1以上のインターフェースをサポートし、それぞれは幾つかのメソッドを有している。メソッドは、特定のアクションを実行する関数や処理であり、オブジェクトを用いたソフトウェアの他の部分（このオブジェクトのクライアント）によって呼び出されることができる。各インターフェースを取りまとめるメソッドは、ある予め決定された手法内にお互いに関連づけられている。クライアントは、オブジェクトのインターフェース内のメソッドの起動のみによって、1つのOLEオブジェクトによって提供されたサービスをアクセスすることができ、クライアントは直接にそのオブジェクトのデータのいずれもアクセスすることができない。

【0013】図の手法によって、この発明による針ゲージ制御は、OLE制御オブジェクトとして実装されている。このOLE制御オブジェクトは、SetMaxTickAngleやGetMaxTickAngleのようなメソッドを含むインターフェースをサポートし、SetMaxTickAngleは最大チック角、すなわち針の最大変位、を定義するメソッドである。一方、GetMaxTickAngleはパラメータとして最大チック角を返すメソッドである。もし、後日、オブジェクトの開発者が、この同じオブジェクトに対してズームングをするサポートをしたい場合、オブジェクトは、拡大要素を特定する角をもった他のインターフェース（おそらく、ZoomInのような単一のメソッドをもった）をサポートする必要がある。各インターフェースのメソッドは、角を表示し、あるいはズームングをアクセスするような、関連するサービスをひとまとめに提供する。

【0014】アプリケーション制御オブジェクトのグラフィカル表示は図1に示されている。オブジェクト30は、このオブジェクトの回りの矩形で示すように、サーバ32の内側にいつも実装されている。このサーバは動的リンクライブラリ（DLL）でもあり、DLLは、アプリケーションが実行され、あるいはそれ自身の分離された処理であるときに必要とされてロードされる。オブ

ジェクト30は、グラフィカルアプリケーション制御オブジェクトのインターフェース内のメソッドを起動するのに用いられる幾つかのインターフェース34が提供されている。オブジェクトは典型的に幾つかの他のインターフェースを介してサービスを提供し、クライアントは、そのインターフェースのメソッドを起動することが計画されている該各インターフェースへの分離されたポインタを持たなければならない。例えば、針ゲージ制御オブジェクトのクライアントは、オブジェクトのチック角インターフェース内のメソッドを起動する、あるインターフェースポインタと、オブジェクトのズームイングインターフェース内のメソッドを起動する他のポインタを必要とする。図2は、2つのメソッド(SetMaxTickAngle, GetMaxTickAngle)をもつクライアント40と、2つのインターフェース34をもつ針ゲージ制御オブジェクト42を示す。インターフェース34の一つは、クライアント40へのポインタ44をもち、これにより、針ゲージ制御オブジェクト42をクライアントのメソッド(SetMaxTickAngle, GetMaxTickAngle)を起動できる。

【0015】全てのグラフィカルアプリケーション制御オブジェクトは、特定クラスのインスタンスである。例えば、あるクラスは、針ゲージの形で機能的にモニターすることを提供するオブジェクトを含み、他のクラスは、アナログバーの形で情報を表示するオブジェクトを含む。各クラスは、そのクラスのオブジェクトによって使用される種々のデータ定義の全てを含む分離されたDLLをもつ。

【0016】種々のサービスへの分離されたアクセスを提供するこのモデルのグラフィカル表示は図3に示される。クライアントが実行するオブジェクト上で所望のインターフェースへのポインタをもつと、そのインターフェース内のメソッドが起動することによってのみでオブジェクトのサービスを用いて開始される。プログラマからみれば、メソッドの起動は、実際、ライブラリの中で実行され実際に実行されるコード、分離された処理、オペレーティングシステムの一部、あるいは全く異なったシステム上であっても、ローカルな処理あるいは関数を起動するように見える。こうして、図3の例では、アプリケーション50、52はともにオペレーティングシステム54上で実行される。アプリケーション50は2つのポインタ56、58を有し、それぞれメソッド60、62をポイントする。さらに、アプリケーション50は、オペレーティングシステム70上で実行されるアプリケーション68内のメソッド66へのポインタ64を有する。このグラフィカル・アプリケーション・デザイン・アーキテクチャを用いると、全ては同様な手法にによってアクセスされるので、クライアントは、これらの区別に気を付ける必要はない。

【0017】この発明の主な目的の一つは、存在するコードの効果的な再利用を組み入れるオブジェクト指向ア

プローチを利用したビル自動制御システムを提供することである。よく定義されたインターフェースおよびメソッドをもって再利用可能な要素(グラフィカル・モニタOLE制御)の生成を許すことによって、新しいデザイン・アーキテクチャは、これを可能にする下部構造を提供する。

【0018】従来のオブジェクト指向技術は、現コードを再利用するための基本的な機構として、(新しいオブジェクトが現オブジェクト内でメソッドの実際の実行を承継する)承継の実行に頼る。この種の承継は、基本オブジェクト内の変更が、実行を承継するオブジェクト上で予期しない結果をもつため、非常に異種のプログラミング環境では、オブジェクト指向システムには非実用的である。例えば、基本オブジェクトが、基本オブジェクトから承継したオブジェクトの独立を更新し、あるいはメモリから削除する状態は、予期しない結果を生じるかもしれない。それに対して、このグラフィカル表示システムはそのような承継に頼ることがなく、その代わり、概念記述された抑制を通した再利用を提供する。

【0019】この発明の抑制は、オブジェクト間の関係に頼って再利用を提供する。外部オブジェクトは、内部オブジェクトのサービスを再利用する一つである。従って、外部オブジェクトは単に内部オブジェクトのクライアントとして動作する。図4に示すように、外部オブジェクト72は、それ自身の関数を実行するため、内部オブジェクト74のメソッドを起動する。しかしながら、外部オブジェクト72は、これらのメソッドをクライアントに見せない。その代わり、クライアント76が外部オブジェクトのインターフェース34の一つのメソッドを起動するとき、このメソッドの実行は内部オブジェクト74のインターフェース34のメソッドに対する呼び出しを含む。換言すれば、外部オブジェクトのインターフェースは、内部オブジェクトのメソッドを呼び出すメソッドを含む。

【0020】制御オブジェクトの使用は、クリックする種々のボタン、ドラッグするスライダ、書き込むためのテキストボックス等を表示する典型的なグラフィカル・ユーザ・インターフェースのみを見るエンドユーザに透過である。WindowsNTのような殆どのオペレーティングシステムは、アプリケーションに対し、ユーザへのこの種のインターフェースを表示するようにしている。従来の技術あるいはデザインアーキテクチャでは、これらのユーザインターフェース特徴をもって表示し、動作するために必要なコードは、システムあるいは特定のアプリケーションの集積部分である。これに対し、この発明のグラフィカル・アプリケーション・デザイン・アーキテクチャによると、ユーザインターフェース特徴はシステムの集積部分ではない。それゆえ、ユーザインターフェースは、制御システムの開発で、分離され、あるいはパラレルで設計される。しかしながら、ユーザからすれ

ば、インターフェースおよび制御システムは全体として集積されて継ぎ目なく連続している。特に、ユーザは、幾つかのOLE制御（以下詳細に後述する、アナログバー、ポイント情報ブロック、針ゲージ、ハイパーリンク、および関連するポイント制御のような）を含む制御内容を見る。制御内容は、OLE要素ドキュメント内容に類似しているが、OLE制御とともに動作する幾つかの外部インターフェースをサポートしている。各制御は、その内容（Container）にプラグされ、それぞれは共通に、正規起動をサポートするはめ込みオブジェクトとして、それ自身のユーザインターフェースを表示する。例えば、スクリーン上のスライダは、その角に沿って動かすことによって、特定の出力ポイントを新規の値にするのに用いることができる。スライダを動かすことによって、操作者は、実際にOLE制御オブジェクトをトリガするコードと相互作用する。エンドユーザが単一のアプリケーションとして見る集積されたユーザインターフェースは、実際、種々に分散されたモニタOLE制御にある制御内容であり、分散されたモニタOLE制御のそれぞれは、完全に分解された部分を提供している。

【0021】内容は、ユーザに対して、制御プロパティを調べ、修正するのに直接的、直観的な方法を提供する。図示した方法によると、図5は、ユーザに制御のプロパティを直接に見せ、修正することを可能にするプロパティウィンドウと呼ばれるユーザインターフェース82を示している。しかし、制御内容の全てはこの種のアクセスには提供されない。さらに、各制御ディスプレイがそれ自身特有のユーザインターフェースを提供させるというよりはむしろ、プロパティページの考えが用いられる。プロパティページをもつと、どんな制御のプロパティも、標準ユーザインターフェースを用いた標準の方法で調べられ、修正される。プロパティウィンドウインターフェース82は、設計するのがかなり複雑であるが、直観的に把握するのが容易である。タブされたダイアログボックスの各ページはそれ自身のプロパティページオブジェクトによって提供される。スタイルプロパティページは図5に示される。

【0022】プロパティページオブジェクトはOLEオブジェクトであり、Iプロパティページインターフェースをサポートするそれ自身のクラスID（CLSID）を全て持っている。CLSIDは、OLEへの特定のサーバ実装を識別する128ビットのユニークな数であり、Iプロパティページインターフェースは、オブジェクトに対してOCX制御のために実行されなければならない標準OLEインターフェースである。

【0023】内容と制御オブジェクトとの相互作用は、図6を参照して以下説明する。図に示すように、制御オブジェクト88は、内容92にそれをサポートするプロパティページオブジェクト94、96について学ばせることを許すI特定プロパティページインターフェース

（破線矢印90で示す）を実装する。ユーザが制御プロパティを見ることを求めると、制御内容92は、その制御をサポートする各プロパティページオブジェクトの一つである、CLSIDのリストを得るメソッドを呼ぶ。

【0024】内容92が、プロパティページ94、96のいずれを制御がサポートするかを知ると、内容92はプロパティフレーム98を生成し、プロパティフレーム98は、コクリエートインスタンスメソッドを用いて各プロパティページオブジェクトのインスタンスを起動する。各プロパティページオブジェクトに対して、プロパティフレームは、ページ位置オブジェクト100、102を提供し、そのそれぞれは、Iプロパティページ位置インターフェースをサポートする。このインターフェースを用いて、プロパティページオブジェクトはそれを生成したプロパティフレームについて学ぶことができる。各プロパティページオブジェクトはプロパティフレームへのページを表し、プロパティフレームはそれらを図5に示すもののようなタグされたダイアログボックスの中に組み立てる。

【0025】オブジェクトのこの構造を用いると、ユーザは直接に制御のプロパティを調べ、修正することができる。どんな変更も、Iプロパティページを介して、プロパティフレームからプロパティページオブジェクトに通信され、その後プロパティページオブジェクトによって、Iディスパッチインターフェースを介してその制御自身に返される。さらに、グラフィック制御内容は、全表示環境内に制御を集積するのを容易にするため、すなわち新しい制御がグラフィック内容内に挿入される場合、のプロパティを備えている。例えば、挿入された制御は、現背景色に適合し、それ自身のテキストを内容のデフォルトフォント内に表示させ、あるいは動的（実行時間）に編集モードか否かを決定する。

【0026】制御をそれ自身が見いだす環境について学ばせることを許容するため、グラフィックス制御内容は環境プロパティをサポートし、この環境プロパティはデフォルト背景色、デフォルトフォント、モードなどを含んでいる。制御は、Iディスパッチインターフェースを介してこれらのプロパティについて学ぶことができる。内容の環境プロパティの値を一度得ると、制御はこれらの値にそれ自身のプロパティを修正することができ、視覚的に内容内の他の制御を集積することを許容する。

【0027】関連するオブジェクト制御を用いた制御ポイントのモニタの方法は図7-9を参照して説明される。特に、図7は関連するオブジェクト制御を図示し、図8はこの発明に従った関連するオブジェクト制御を生成するフローチャートである。

【0028】この発明に従った関連するオブジェクト制御はいくつかの独立した選択可能なオブジェクトを含む複合グラフィックである。この複合グラフィックは、絵画的な形式で、システムあるいはシステムの一部を示

し、直観的な方法で状態情報を動的に表示する。図によれば、図7はビル自動システム内における空気取扱システム110の一部を示し、幾つかの独立した選択可能なオブジェクト112, 114, 116を含んでいる。

【0029】この発明の方法を用いると、ユーザは選択したオブジェクトを所望の制御ポイントに関連づけ、制御ポイントの現状態を可視表示する種々のスタイル特性を選択する。この方法は、図8を参照して以下説明する。

【0030】まず、ユーザは、格納された複合グラフィックのライブラリから一つの複合グラフィックを選択する(ブロック122)。次に、ユーザは、例えばキーボード、マウス、接触感知スクリーン、あるいは他のデータ入力方法を用いる、独立した選択可能なオブジェクトの一つを選択する(ブロック124)。

【0031】その後、ユーザは、制御ポイント内への入力、あるいは定義された制御ポイントのリストから一つの制御ポイントを選択することによって、選択されたオブジェクトに関連づける(ブロック126)。制御ポイントを入力し、あるいは選択する操作者インターフェースは、例えば図9に示される。この発明に従ったポイント名は、階層的にユニークなポイントを記述した1以上のデリミネータによって分けられたアルファベット数字文字の文字列である。図9に示すインターフェースを用いると、ユーザは全ポイント名を選択的に表示させるために選ぶことができ、ユーザはデリミネータを特定でき、最初のおよび最後のフィールドの一つを表示させるために選ぶことができる。

【0032】図8を再び参照すると、ユーザは、設定ポイントおよび1以上のオフセットを特定することによって、関連づけられた制御ポイントに対する状態を指示する(ブロック128)。この情報の入力に対する操作者インターフェースは例えば図10に示す。設定ポイントは制御ポイントに対する所望のあるいは通常の状態の中央ポイントを指示する。第1アラーム状態はオフセットの入力によって特定される。例えば、与えられた制御ポイントが50で第1オフセットが10である場合、通常状態は40から60までの範囲の値を含む。この範囲を逸脱するどんな値もアラーム状態がトリガーされる。さらに、ユーザは、追加のオフセットを入力することによって追加のアラーム状態を特定する選択をする。

【0033】再び図8を参照すると、ユーザは次に、制御ポイントが仮定された状態のそれぞれを表示するために用いる状態色を選択する(ブロック130)。これにより、通常状態は、例えば、緑色によって指示され、第1アラーム状態はオレンジ色によって指示される。

【0034】さらに、ユーザは、制御ポイントがアラーム状態に入ったときのグラフィックオブジェクトおよび／あるいはポイント名をブリンクさせることを選ぶことができる。最後に、ユーザは、関連づけられたオブジェ

クトの状態が、特定された状態特性に従って表示されたオブジェクトをリフレッシュすることによって動的に更新されるシステムに警戒させる(ブロック132)。

【0035】この発明の重要な特徴は、ユーザが、グラフィックオブジェクトと制御ポイントとの間のリンクを追加し、変更し、削除することができることである。これは、オブジェクトを制御ポイントにリンクすることを記述した同一の処理を用いて達成される。これにより、モニタされる追加の新しい制御ポイントは、管理制御システムの再プログラミングの要求なしに、達成される。

【0036】他の重要な特徴は、各状態に関連づけられた表示特性は、ユーザが、記述された操作者インターフェースを用いた個々の選択に適するように容易にカスタマイズされる。

【0037】この発明の他の観点によれば、複合グラフィックは、静的背景図上をオーバーレイすることができる。例えば、環境制御装置を示すグラフィックは、ビルの静的レイアウト上をオーバーレイすることができる。この特徴は、大きなシステム内の制御配置をユーザにグラフィカルに示すことを提供し、美しい背景を提供する、現在のビットマップ・グラフィックスの使用を容易にする。

【0038】静的背景図を含む動的制御の生成のためのデータとコマンドのフローは、図11を参照して説明される。特に、データベース150は、1以上の静的背景図152、および1以上の独立した選択可能なグラフィック制御オブジェクトを含む1以上の複合グラフィック154を生成するデータを有する。処理ステップ156(矢印で示す)において、操作者158は、データベース150に格納された図の中から静的な図を選択することによって、背景グラフィックを生成するように選ぶ。次の処理ステップ(ブロック160)では、操作者は背景図をオーバーレイする所望の複合グラフィックを選択し、独立して選択可能なオブジェクトの選択されたものを所望の制御ポイントに関連づけることによって、動的グラフィックを生成する。さらに、操作者は、各関連づけられたオブジェクトに対するアラームや表示特性を含む所望の動的特徴を選択する。最後の処理ステップ(ブロック162)において、操作者158は、動的グラフィックを用いた関連する制御ポイントのものをを選択する。

【0039】この発明のさらに他の観点からすれば、各独立した選択可能なグラフィックオブジェクトは、1以上の動的なサブオブジェクトを複合した分離OCX制御である。分離OCX制御としての各グラフィックオブジェクトの実行は、モジュラ制御の開発を容易にする。さらに、この発明のこの観点は、個々のグラフィックオブジェクトがリフレッシュされ、背景と他のグラフィックオブジェクトとは無関係に、高速リフレッシュ応答を可能にする。

【0040】この発明のグラフィカル表示方法の第2実施例は、図12-15を参照して以下説明する。端的に言えば、この発明の第2実施例に従った実行時間値および状態情報を動的に表示するのに用いられる3つの制御がある。すなわち、針ゲージ制御、アナログバー、およびポイント情報ブロックである。この発明に従った針ゲージ制御は図12に示され、アナログバー制御は図13に示され、ポイント情報ブロック制御は図14に示される。

【0041】これらの制御は、幾つかの点で従来のビル自動制御から区別される。これらの点の一つは、制御のモジュラリティに関係する。特に、この発明の各制御は、個々のデータファイルに格納された分離OCXである。ユーザは、制御がモニタされることを選び、制御ポイントがモニタされる制御のタイプを特定することができる。さらに、制御の各インスタンスは、他の制御とは独立し、その処理サイクルの間、必要によって各制御がリフレッシュされる。換言すれば、この発明のOCX制御は、マルチスレッド環境で、独立的にかつ動的に更新される。

【0042】特に、各制御は、静的および動的なサブオブジェクトを複合したオブジェクトである。静的サブオブジェクトは、リフレッシュ期間にリフレッシュされず、あるいは再描画されないオブジェクトの部分である。図によれば、図13の針ゲージ制御168では、スケール170と制御ポイント名172が静的サブオブジェクトである。これに対して、矢印セグメント174と矢印ヘッド176は、動的サブオブジェクトであり、これらは、制御ポイントの現実実行時間値を反映して動的にリフレッシュされるからである。この発明による、動的サブオブジェクトから静的サブオブジェクトを独立することは、各処理サイクル要求の間で処理リソースが動的サブオブジェクトをリフレッシュするのに用いられるような、高速マルチスレッド動作を容易にする。

【0043】動的制御を生成する方法は図15を参照して以下説明する。この方法は、関連づけられたオブジェクト制御を生成する方法に非常に似ており、主な違いは、個々の制御に関連するカスタマイズ可能なオプションがあることである。

【0044】ユーザは、例えば針ゲージ、アナログバー、ポイント情報ブロックを含むグループの中から針ゲージを選択する（ブロック182）。次に、ユーザは、その制御にリンクされ、あるいは関連づけられるポイントを選択する（ブロック184）。

【0045】その後、選択された制御に対するスタイル特性は特定される（ブロック186）。ユーザが、スタイルを修正し、制御の各タイプのオプションを表示することができることは、この発明のもう一つの重要な特徴である。これに対して、従来のビル自動システムにおけるグラフィカル制御は、種々のスタイルや表示特性がグ

ラフィカル表示プログラム内にコード化されていたので、柔軟性がなかった。すなわち、従来の制御のサイズ変更には、プログラム変更が要求された。これに対して、この発明による制御のスタイルおよび表示特性はプログラム変更の要求なしに、ユーザによって修正することができる。

【0046】例えば、図5は針ゲージのスタイル特性を設定する操作者インターフェース82を示す。この操作者インターフェース82を用いると、ユーザは、針の矢印ヘッドおよび／または旋回中心点を示すかを選ぶ。さらに、ユーザは、旋回方向と旋回中心位置、同様に最大および最小の針偏差（度）を特定する。

【0047】これに対して、選択された制御がアナログバーである場合、ユーザは、バーの方向を選択することができる（ブロック186）。例えば、図16は、水平方向のアナログバーあるいは垂直方向のアナログバーを選択することができる操作者インターフェース200を示す。さらに、ユーザは、バーのデフォルト方向を選ぶことができる。すなわち、水平方向のバーに対しては、ユーザは、バーが左あるいは右のいずれに増大するのかが選択し、垂直方向のバーに対しては、ユーザは、バーが上あるいは下のいずれに増大するのかが選択する。

【0048】さらに、制御のタイプがポイント情報ブロックである場合、ステップ186において、ユーザは操作者インターフェース210を用いて種々のスタイル特性を選択することができる（図17）。他のものの間では、ユーザは、関連する制御ポイントの名前の表示、ディスクリプタ、そのステータス、およびそのアラーム優先度を選ぶことができる。

【0049】その後、ステップ188において、ユーザは制御ポイントに対する状態を定義し、各状態に対応するユニークな色を特定する。すなわち、制御ポイントに対する実行時間値を動的に表示することに加え、状態情報は、各状態に関連づけられた色を用いて表示される。

【0050】ユーザは、さらに、選択された制御に対する表示オプションを選択する。例えば表示すべきタイトル、関連づけられたポイント名とアラームマークを表示するか否か、同様に軸ラベルおよび関連づけられたフォントを特定することである（ブロック190）。次に、ユーザは、制御ポイントが仮定された各状態を表示するのに用いられる状態の色を選択する（ブロック192）。

【0051】最後に、ポイントにリンクされた状態は、特定されたアラーム特性と実行時間値とに応じて動的サブオブジェクトをリフレッシュすることによって動的に表示される（ブロック194）。

【0052】上述したように、ユーザは、制御の各タイプに対するアラーム状態特性を特定する能力をもつ。例えば、ユーザは、針偏差あるいはアナログバーの配置を介して実際の実行時間値を示しつつ、制御ポイントのア

ラーム状態を応じた色に、制御の色を変更することを特定することができる。

【0053】上述したことから、グラフィカルに、リアルタイム処理状態情報を表示する再利用可能な制御オブジェクトを用いるビル自動システムでの、動的処理情報をモニタし、制御するモジュラグラフィカル表示方法が、示され、記述されたものとする。さらに、このモジュラグラフィカル表示方法を用いたビル自動制御システムが示され、記述されたものとする。

【0054】一方、この発明の種々の実施例は示され、かつ記述されたので、他の修正、置換、選択は、先行技術によって明らかであると理解される。そのような修正、置換、および選択はこの発明の思想および観点から分離されずに行うことができ、この発明は、付加された請求の範囲から決定される。

【0055】この発明の種々の特徴は、付加された請求の範囲で明らかにされる。

【図面の簡単な説明】

【図1】この発明によるグラフィカル・アプリケーション・制御オブジェクトを示す。

【図2】針ゲージ制御オブジェクトへのインターフェースを持ったクライアントを示す。

【図3】この発明に従った種々のソフトウェアによって提供されるサービスをアクセスするのに用いられる標準モデルを示す。

【図4】この発明に従った抑制を通したオブジェクトの

再利用の一例を示す。

【図5】制御のプロパティを修正するためのユーザ・インターフェースを示す。

【図6】制御のプロパティを修正するためのフローチャートを示す。

【図7】関連するポイント制御を示す。

【図8】この発明の第1実施例に従った制御ポイントをもつグラフィカルオブジェクトに関連づける方法のフローチャートを示す。

【図9】グラフィカルオブジェクトに関連づけられたポイント名を入力するための操作者インターフェースを示す。

【図10】グラフィカルオブジェクトに関連づけられた制御ポイント名の状態情報を入力するための操作者インターフェースを示す。

【図11】静的背景図上に載る動的制御を生成するためのデータおよびコマンドの流れを示す図である。

【図12】針ゲージ制御を示す。

【図13】アナログバー制御を示す。

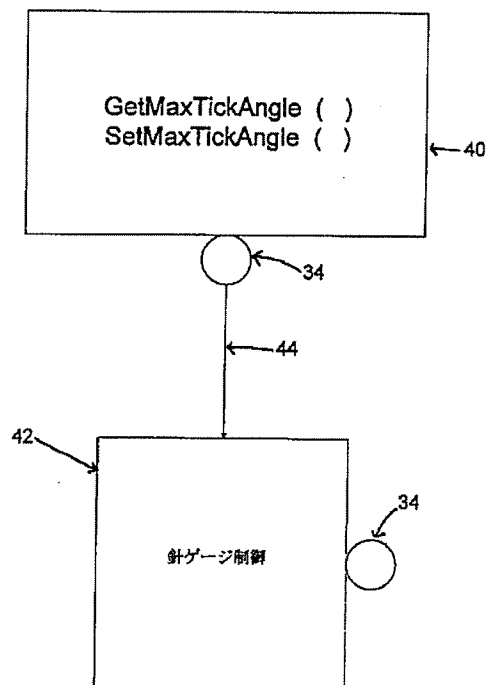
【図14】ポイント情報ブロックを示す。

【図15】この発明の第2実施例に従った動的制御を生成するためのフローチャートを示す。

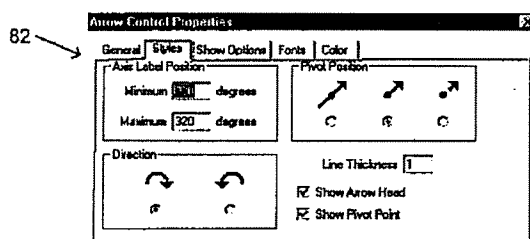
【図16】アナログバー制御のスタイル情報を特定する操作者インターフェースを示す。

【図17】情報ブロック制御のスタイル情報を特定する操作者インターフェースを示す。

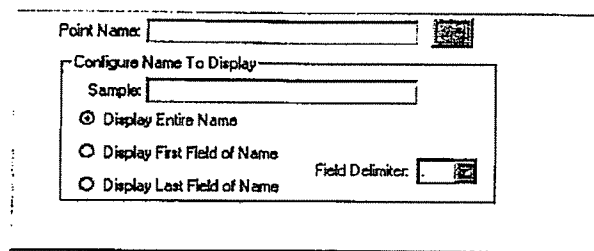
【図2】



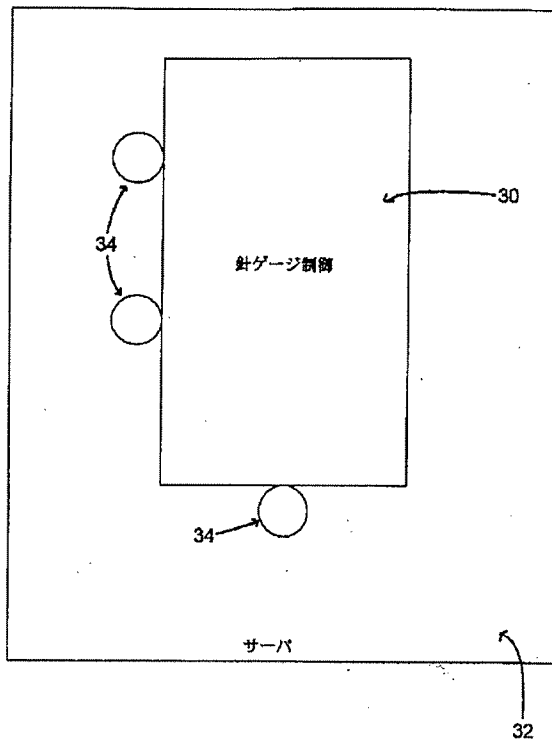
【図5】



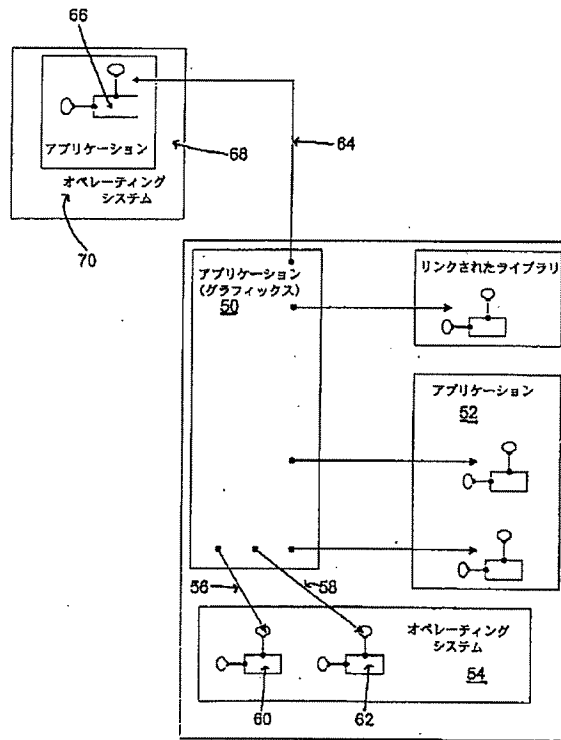
【図9】



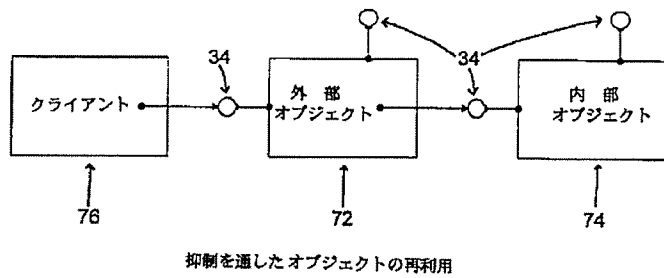
【図 1】



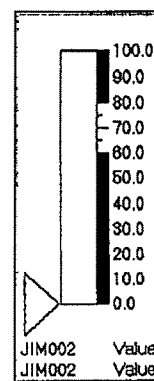
【図 3】



【図 4】



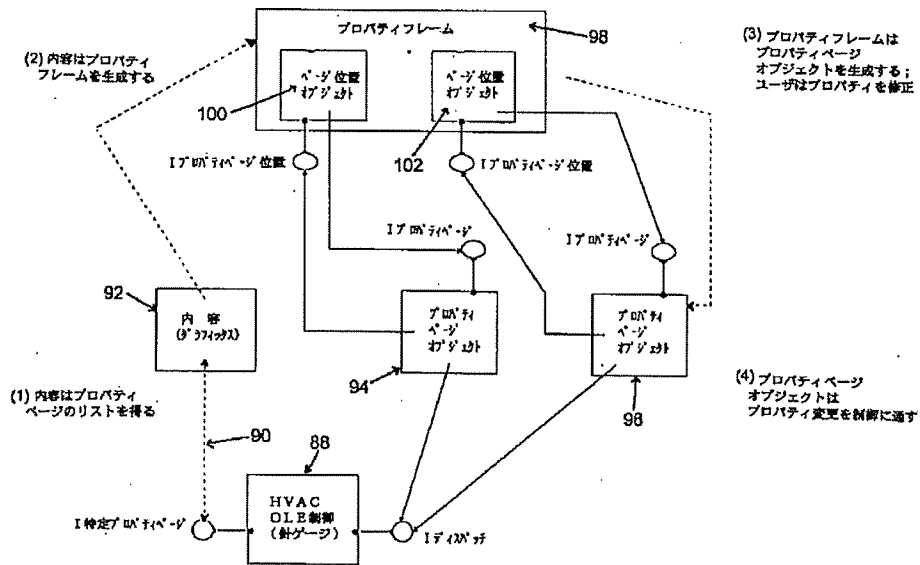
【図 13】



【図 14】

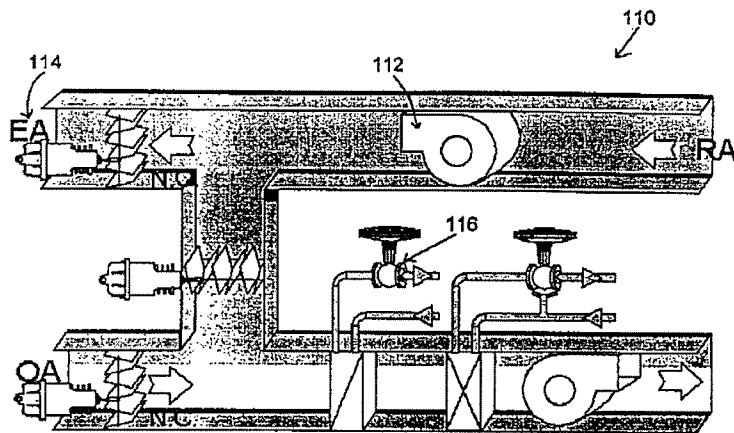
JIM002U JAMES POINT Status Priority Value	JIM003 TEST POINT Status Priority Value
---	---

【図6】

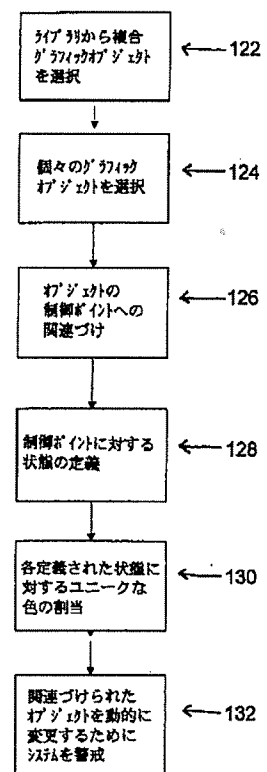
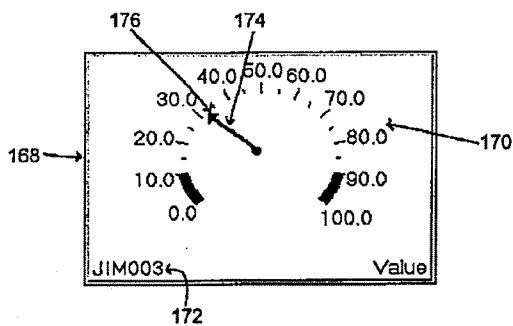


【図7】

【図8】



【図12】



【図10】

Enhanced Alarm Setup

Alarm Destinations: [001 Dest] [None] [None] [None]

Mode Point: [LA001]

Mode Delay: [0] (min.)

Level Delay: [0] (sec.)

Differential: [0]

☐ Acknowledge Return to Normal

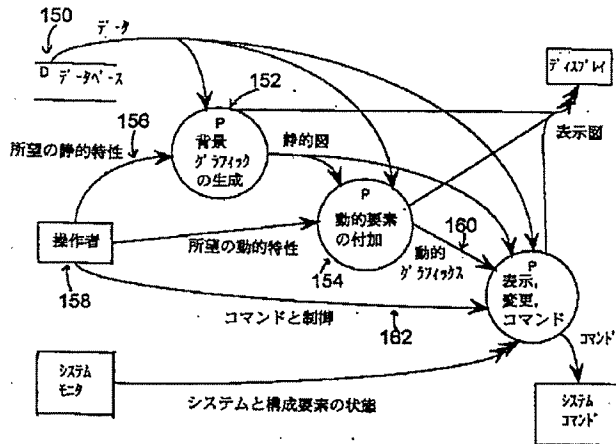
Day: [Night] [Special 2] [Special 3] [Special 4] [Special 5]

☒ Alarm Mode Enabled

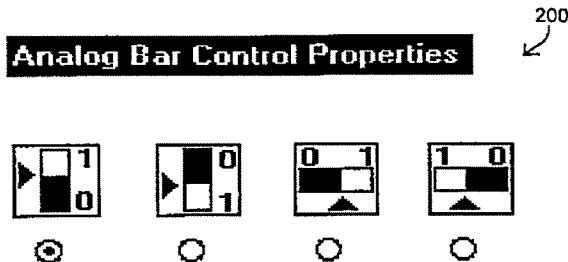
Set Point: Name: [] Value: [50]

Offset	Priority	Extra Destination	Enhanced Alarm Message
1) 10	PRI1	[None]	[None]
2) 20	PRI2	[None]	[None]
3) 30	PRI3	[None]	[None]
4) 40	PRI4	[None]	[None]

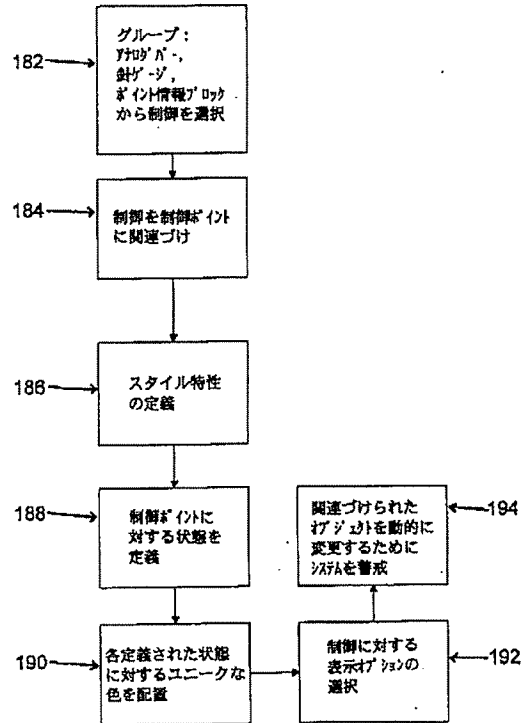
【図11】



【図16】



【図15】



【 17 】

210

Fields

- ☒ Display Name
- ☒ Descriptor
- ☒ Status
- ☒ Priority
- ☒ Current Value ☒ Units
- ☐ Totalized Value

Display Style

☒ Multiple Lines ☐ Single Line

State Color

☒ Use For Background Rectangle

☐ Use For Font Color

☐ Use Reverse Video Instead

☒ Border

1. Title of Invention

METHOD AND APPARATUS FOR MONITORING AND CONTROLLING
REAL-TIME INFORMATION IN A BUILDING AUTOMATION SYSTEM

2. Claims

1. A modular graphical display method for monitoring and controlling dynamic process information in a building automation system, comprising:
selecting a displayed graphic object from among a plurality of independent graphic objects;
linking said selected object with a control point;
defining a normal state and at least one alarm state for said control point;
assigning a state characteristics corresponding to each of said defined states;
and
dynamically displaying the state of said control point using said alarm characteristics.
2. A modular graphical display method according to claim 1 wherein said state characteristics are selected using a state color table.
3. A modular graphical display method according to claim 2 wherein said alarm characteristics include displayed text and blinking said graphic object.
4. A modular graphical display method according to claim 1 wherein said graphic object is overlaid over a static background picture.
5. A modular graphical display method according to claim 1 wherein each said independent graphic object is an OCX control.
6. A modular graphical display method for monitoring and controlling dynamic process information in a building automation system, comprising:

selecting a displayed graphic object from the group comprising an information block control, an analog bar control, and a gauge needle control, said selected graphic object comprising at least one static sub-object and at least one dynamic sub-object;

specifying style characteristics for said selected graphic object;

linking said selected object with a control point;

defining a normal state and at least one alarm state for said control point;

assigning state characteristics to each said defined state; and

dynamically displaying run time values and state information of said control point by refreshing said dynamic-sub-objects in accordance with said state characteristics.

7. A modular graphical display method according to claim 6 wherein said graphic object is an analog bar and includes a caret for commanding said linked point, said caret being a dynamic sub-object.

8. A modular graphical display method according to claim 7 wherein said caret is moveable only if a user profile indicates that a user may command said linked point.

9. A modular graphical display method according to claim 7 wherein said analog bar has user adjustable attributes including bar orientation, size, scale, and alarm limit.

10. A modular graphical display method according to claim 6 wherein said graphic object is capable of being navigated to a point commander for commanding the state of said point.

11. A building automation control system, comprising:

- at least one environmental control device;
- a supervisory control station including a CPU, memory means for storing data, communications means for sending and receiving data to and from said environmental control device, display means for displaying graphical information, and an input unit, said supervisory control station receiving real-time process state information for at least one control point from said environmental control device;
- at least one control object for graphically displaying said real-time process state information for a selected said one control point;
- said memory means storing a dynamic link library (DLL) for each said at least one control object, said DLL containing:
 - first data for defining each of plural services as one or more OLE objects, each said OLE object supporting one or more interfaces, each said interface including a number of methods, said methods being callable only by invoking said OLE object's interface;
 - second data for displaying at least one independently selectable graphical object;
 - link data specifying an association between said graphic object and said control point;
 - a state table containing user selectable change of state information for graphically displaying a change of state for said control point; and
 - style data specifying optional, user selected preferences including at least one of displayed fields, position, and size;
- wherein said control station dynamically displays said received real-time process state information for said control point using said state table data, said link data and said first data.

12. A building automation control system according to claim 11 wherein each said graphic object is comprised of graphic sub-objects selected from the group of static sub-objects and dynamic sub-objects, said dynamic sub-objects being capable of being refreshed independent of said static sub-objects;

said state table data including data for each state which said dynamic sub-objects can assume.

13. A building automation control system according to claim 11 further comprising:

data for displaying a static background graphic stored in said memory;

wherein said display means displays at least one graphic object overlaid on said static background graphic, said graphic object being addressed and refreshed independent of said static background graphic.

14. A building automation control system according to claim 11 further comprising:

hyper-link data specifying an associated graphic object;

wherein said display means displays at least one graphic object overlaid on said static background graphic, said graphic object being addressed and selected independent of said static background graphic.

15. A building automation control system according to claim 11 wherein said control object is an analog control, and said DLL further comprises:

analog bar style data specifying the orientation of said analog control, alarm marks, axis labels, point name, and high and low tick numbers specifying the displayed range of values for said control point.

16. A building automation control system according to claim 11 wherein said control object is a needle gauge control, and said DLL further comprises:

analog bar style data specifying the orientation, size and position of said needle gauge control.

17. A building automation control system according to claim 16 wherein said DLL further comprises data specifying a selected needle style from among a plurality of needles styles.

3. Detailed Description of Invention

BACKGROUND OF THE INVENTION

The present invention relates to a method and apparatus for monitoring and controlling real-time information in a building automation system. More specifically, the present invention discloses a flexible, modular apparatus for dynamically displaying real-time process information. The present invention also discloses a uniform approach by which one section of code can access the services of another section of code, thereby promoting reuse of code and ease of maintenance.

Conventional systems for monitoring and controlling graphical displays are inflexible and utilize graphical controls or images which are rigidly linked or associated with specific control points during the coding of the user interface. In this context, a control point is a physical point connected to a sensor or apparatus such as an environmental control device which can either be monitored or commanded. Because the linkage or association is coded into the interface, these prior art systems are inflexible, and the subsequent addition of further control points requires significant program modifications. This inflexibility is a serious drawback which adds to the cost of upgrading a building automation system.

An additional drawback associated with prior art graphical display systems is that the various attributes associated with these graphical controls or images are fixed. Thus, changes to the size of a control, font, scale or the like could only be achieved by modifying and recompiling the underlying program. From a user perspective, this inflexibility is highly undesirable.

An additional drawback associated with prior art graphical display systems relates to the manner in which these systems access the software services provided by another piece of software, application, or library. A conventional graphical monitoring application would link to a library and then access the library's services by calling the functions in the library. Alternatively, the graphical display application would use the services provided by another application, which runs in an entirely separate process. In this scenario, the two local processes would communicate using an inter-process communication mechanism, which requires a protocol between the two applications (a set of messages allowing one application to specify its requests and the other to respond appropriately). In yet another scenario, the graphical display application uses the services provided by the operating system, i.e., the application makes a series of system calls, each of which is handled by the operating system.

Due to the lack of any standard approach for sharing services, a conventional graphical monitoring application may include any number of these different approaches to access the services provided by another. In turn, the use of these disparate approaches inhibits the reuse of code sections and makes it difficult to maintain the software. Thus, a single, uniform approach was needed for providing one code section with access to the services of another.

SUMMARY OF THE INVENTION

Accordingly, in response to the problems discussed above, one object of the present invention is to provide an improved building automation control system that has a set of flexible, modular controls that enables a user to add, change or delete an association or link between a graphic object and a control point without requiring modification and/or recompiling of the underlying programming.

Another object of the present invention is to provide such a system wherein a user can modify attributes associated with a graphic object without requiring modification and/or recompiling of the underlying programming.

Another object of the present invention is to provide such a system that includes a graphical monitoring application including a uniform approach by which one section of software supplies its services to another section of software, thus promoting reuse of code sections, ease of development and maintenance of the software.

These and other objects of the present invention will be apparent from the following detailed description of the invention, while referring to the attached drawings.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Broadly stated, the above objects are met or exceeded by the present modular graphical display method for monitoring real-time information in a building automation system. According to the method of the present invention the user selects a graphic object to be displayed from among a plurality of independent graphic objects, and links the selected object with a control point. Next, the user defines a normal state and at least one alarm state for the associated control point, and assigns state characteristics corresponding to each of the

defined states. Finally, the state of the control point is dynamically displayed on the supervisory control station using the alarm characteristics.

The above objects are also met or exceeded by the apparatus of the present invention for monitoring and controlling real-time information in a building automation system. Generally, in the preferred embodiment, the present invention is a building automation control system that includes a supervisory control station for monitoring and controlling process information (control points) received from at least one environmental control device. The supervisory control station preferably includes a CPU, communications apparatus for sending and receiving data to and from the environmental control device, a display for displaying graphical information, and an input unit. The supervisory control station receives real-time process state information for a selected control point from the environmental control device via the communications apparatus. Moreover, the real-time process state information is dynamically displayed using a control object. The supervisory control station is provided with a memory for storing data, including a dynamic link library (DLL) for each control object. The DLL for each control object contains first data for defining each of plural services as one or more object oriented linking and embedding (OLE) objects, each OLE object supporting one or more interfaces. In turn, each interface includes a number of methods, which are callable only by invoking the OLE object's interface. The DLL further contains second data for displaying at least one independently selectable graphical object, and link data specifying an association between a selected graphic object and the control point. The supervisory control station dynamically displays real-time process state information for the associated control point using stored state table data, link data, and run time values from the environmental control device.

According to the graphical application design architecture of the present invention, each section of software implements its services as one or more OLE objects, where OLE refers to the Microsoft Corp.'s object oriented linking and embedding language

(OLE) and is a registered trademark of the Microsoft Corp. Each OLE object supports one or more interfaces, each of which in turn includes a number of methods. A method is a function or a procedure that performs a specific action and is callable by other sections of

software using the object (the client of that object). The methods that make up each interface are related to one another in some predetermined manner. Clients can access the services provided by an OLE object only by invoking the methods in the object's interface, clients cannot directly access any of the object's data.

By manner of illustration, a needle gauge control according to the present invention is implemented as an OLE control object. This OLE control object supports an interface which includes methods such as SetMaxTickAngle, and GetMaxTickAngle, where SetMaxTickAngle is a method for defining a maximum tick angle, i.e., the maximum displacement of the needle. In turn, GetMaxTickAngle is a method which returns the maximum tick angle as a parameter. If at a later date the object's developer desired to add support for zooming to this same object, the object would need to support another interface (perhaps with a single method such as ZoomIn) with an argument specifying the magnification factor. The methods in each interface collectively provide related services, such as angle displaying or access to zooming.

A graphical representation of an application control object is illustrated in FIG. 1. The object 30 is always implemented inside a server 32, shown as the rectangle around the object. This server can either be a dynamic-link library (DLL), which is loaded as needed when an application is running, or a separate process of its own. The object 30 is provided with several interfaces 34 which are used to invoke the methods in a graphical application control object's interface. An object typically provides its services through several other interfaces, and the client must have a separate pointer to each interface whose methods it plans to invoke. For example, a client of a needle gauge control object would need one interface pointer to invoke the methods in the object's tick angle interface and another pointer to invoke the method in the object's zooming interface. FIG. 2 shows a client 40 having two methods (SetMaxTickAngle, GetMaxTickAngle) and a needle gauge control object 42 with two interfaces 34. One of the interfaces 34 has a pointer 44 to the client 40,

thereby enabling the needle gauge control object 42 to invoke the client's methods (SetMaxTickAngle, GetMaxTickAngle).

Every graphical application control object is an instance of a specific class. For example, one class may contain objects that provide monitoring functionality in a form of a needle gauge while another class may contain objects representing information in the form of an analog bar. Each class has a separate DLL containing all of the various data definitions used by the objects of that class.

A graphical representation of the present model for providing shared access to various services is depicted in FIG. 3. Once a client has a pointer to the desired interface on a running object, it can start using the object's services simply by invoking the methods in the interface. From the programmer's perspective, invoking a method looks like invoking a local procedure or function whereas in fact, the actual code that is executed might be running in a library, a separate process, as part of the operating system, or even on an entirely different system. Thus, for example in FIG. 3, Applications 50 and 52 are both running on operating system 54. Application 50 has two pointers 56, 58 which point to methods 60, 62 respectively. Moreover, Application 50 has a pointer 64 to a method 66 in Application 68 which is running on operating system 70. Using the present graphical application design architecture, clients need not be aware of these distinctions because everything is accessed in the same manner.

One of the primary goals of the present invention is to provide a building automation control system utilizing an object oriented approach which incorporates an effective reuse of existing code. By allowing the creation of reusable components (graphical monitoring OLE controls) with well defined interfaces and methods, the new design architecture provides an infrastructure that makes this possible.

Conventional object oriented technologies rely on implementing inheritance (in which a new object inherits the actual implementation of methods in an existing object)

as their fundamental mechanism for reusing existing code. This kind of inheritance is impractical for an object oriented system in a very heterogeneous programming environment because changes in base objects could have unexpected effects on objects that inherit implementation from them. For example, a situation in which a base object is either updated or released from memory independent of the objects which inherit from the base object, may cause unexpected results. In contrast, the present graphical display system does not rely on such inheritance, and instead provides for reuse through a concept termed containment.

Containment in the present invention provides for reuse by relying on the relationship between objects. An outer object is one that reuses the services of an inner object. Thus, the outer object simply acts as a client of the inner object. As shown in FIG. 4, the outer object 72 invokes the methods of an inner object 74 in order to carry out its own functions; however, the outer object 72 does not make those methods visible to its client 76. Instead, when the client 76 invokes a method in one of the outer object's interfaces 34, the execution of that method includes a call to a method in an interface 34 of the inner object 74. In other words, the outer object's interface contains methods that call the inner object's methods.

The use of control object is transparent to the end user, who only sees a typical graphical user interface displaying various buttons to click on, sliders to drag, text boxes to fill in, and so on. Most operating system such as Windows NT allow applications to present this kind of interface to the user. In the prior art or design architecture, the code necessary to display and work with these user interface features was an integrated part of the system or of a specific application. In contrast, according to the graphical application design architecture of the present invention, the user interface features are not an integrated part of the system. Therefore, the user interface may be designed separately or in parallel with the development of the control system. However, from the user perspective, the interface and the control system constitute a seamlessly integrated whole. Specifically, the user sees a

control container that includes a number of OLE controls (such as the Analog Bar, Point Information Block, Needle Gauge, Hyperlink, and Associated Point control each of which will be described in detail below). A control container is analogous to an OLE compound documents container, but it supports a few extra interfaces for working with OLE controls. Each control is plugged into the container, and each commonly presents its own user interface as an embedded object supporting in-place activation. A slider on the screen, for example, can be used to command a specified output point to a new value simply by moving it along its axis. By moving the slider, the operator interacts with the code that actually triggers an OLE control object. The integrated user interface which the end user sees as a single application, is in fact a control container populated with various discrete monitoring OLE controls, each providing part of the complete solution.

The container provides a direct, intuitive way for a user to examine and modify a control properties. By manner of illustration, FIG. 5 depicts a user interface 82 termed a Properties window which enables a user to directly view and modify a control's properties. However, not all control containers provide this kind of access. Moreover, rather than letting each control display provide its own idiosyncratic kind of user interface, the notion of property pages is used. With property pages, the properties of any control can be examined and modified in a standard way using a standard user interface. The Properties window interface 82 is intuitively easy to grasp, although fairly complex to design. Each page in the tabbed dialog box is provided by its own property page object. A styles property page is shown in FIG. 5.

A property page object is an OLE object, complete with its own class ID (CLSID) that supports the IPropertyPage interface. The CLSID is a 128-bit unique number which identifies a particular server implementation to OLE, and the IPropertyPage interface is a standard OLE interface which must be implemented in order for the object to be an OCX control.

Interaction between a container and a control object will now be described with reference to FIG. 6. As shown, a control object 88 may implement an *ISpecifyPropertyPage* interface (shown as dashed arrow 90) to allow its container 92 to learn about the property page objects 94, 96 it supports. When a user asks to see a control's properties, the container 92 calls a method which in turn gets a list of CLSID's, one for each property page object the control supports.

Once the container 92 knows which property pages 94, 96 a control supports, the container 92 creates a property frame 98, which in turn activates an instance of each property page object using the *CoCreateInstance* method. For each property page object, the property frame provides a page site object 100, 102, each of which supports the *IPropertyPage Site* interface. Using this interface, a property page object can learn about the property frame that created it. Each property page object presents its page to the property frame, which assembles them into a tabbed dialog box like the one shown in FIG. 5.

Using this structure of objects, a user can directly examine or modify the control's properties. Any changes are communicated from the property frame to the property page objects via *IPropertyPage* and then by the property page objects directly back to the control itself through its *IDispatch* interface. Moreover, a graphics control container may be equipped with properties to facilitate the integration of a control into the overall display environment, i.e., when a new control is inserted into the graphics container. For example, an inserted control may adopt the current background color, cause its own text to appear in the container's default font, or decide whether or not it should be in a dynamic (runtime) or edit mode.

To allow a control to learn about the environment in which it finds itself, the graphics control containers supports ambient properties, which include a default background color, default font, mode, and more. A control can learn about these properties via an *IDispatch* interface. Once it has obtained the values of its container's ambient properties, a

control can modify its own properties to these values, allowing it to visually integrate with the other controls in the container.

A method of monitoring a control point using an associated object control is explained with reference to FIGS. 7 - 9. In particular, FIG. 7 is a representative illustration of an associated object control, and FIG. 8 is a flow diagram for creating an associated object control according to the present invention.

An associated object control according to the present invention is a composite graphic including several independently selectable objects. The composite graphic depicts a system or a portion of a system in pictorial form, and dynamically displays state information in an intuitive manner. By manner of illustration, FIG. 7 shows a portion of an air handling system 110 in a building automation system, including several independently selectable objects, 112, 114, 116.

Using the method of the present invention, a user associates a selected object with a desired control point, and selects various style characteristics used to visually display the present state of the control point. This method is now explained with reference to FIG. 8.

First, a user selects a composite graphic from a library of stored composite graphics (block 122). Next, a user selects one of the independently selectable objects using, for example a keyboard, mouse, touch sensitive screen or other data entry method (block 124).

Then a user associates the selected object by entering in a control point or selecting a control point from a list of defined control points (block 126). An operator interface for entering or selecting a control point is shown, for example, in FIG 9. A point name according to the present invention is a string of alpha-numeric characters separated by one or more delimiters which hierarchically describe a unique point. Using the interface of

FIG. 9, the user can elect to have the entire point name displayed. Alternatively, the user can specify a delimiter and elect to have one of the first and last fields displayed.

Referring again to FIG. 8, the user designates the state(s) for the associated control point by specifying a set point and one or more offsets (block 128). An operator interface for entering this information is shown, for example, in FIG 10. The set point designates the center point of the desired or normal state for the control point. A first alarm state is specified by entering an offset. For example, if for a given control point is 50 and a first offset is 10, then the normal state would encompass values ranging from 40 to 60. Any value falling outside of this range would trigger an alarm state. Moreover, the user elect to specify additional alarm states by entering additional offsets.

Referring again to FIG. 8, the user next selects the state colors used for displaying each of the states which the control point can assume (block 130). Thus, a normal state could, for example, be designated by the color green, and a first alarm state by the color orange.

Moreover, the user can elect to have the graphic object and/or the point name blink when the control point enters an alarm state. Finally, the user alerts the system that the state of the associated object should be dynamically updated by refreshing the displayed object in accordance with the specified state characteristics (block 132).

An important characteristic of the present invention is that the user can add, change or delete the link between a graphic object and a control point. This can be accomplished using the same procedure described for linking an object with a control point. Thus, the addition new control points to be monitored can be accomplished without requiring reprogramming of the supervisory control system.

Another important characteristic is that the display characteristics associated with each state can easily be customized to suit the individual preferences the user using the described operator interfaces.

According to another aspect of the present invention, the composite graphic can be overlaid over a static background picture. For example, a graphic representing an environmental control device can be overlaid over a static layout of the building. This feature facilitates the use of existing bit mapped graphics to provide the user with a graphic representation of the location of a control within a larger system, or provide an aesthetic background.

The data and command flows for creating a dynamic control including a static background picture are now explained with reference to FIG. 11. In particular, a database 150 contains data for creating one or more static background pictures 152 and one or more composite graphics 154 which include one or more independently selectable graphic control objects. In process step 156 (shown by an arrow), an operator 158 elects to create a background graphic by selecting a static picture from among the pictures stored in the database 150. In the next process step (block 160), the operator selects a desired composite graphic to be overlaid on the background pictures, and creates a dynamic graphic by associating selected ones of the independently selectable objects with desired control points. Moreover, the operator selects desired dynamic features including alarm and display characteristics for each associated object. In the final process step (block 162), the operator 158 commands and controls selected ones of the associated control points using the dynamic graphic.

According to yet another aspect of the present invention, each independently selectable graphic object is a separate OCX control composed of one or more dynamic sub-objects. The implementation of each graphic object as a separate OCX control facilitates the development of modular controls. Additionally, this aspect of the present invention enables a faster refresh response, since an individual graphic object may be refreshed, independent of the background and the other graphic objects.

A second embodiment of the graphical display method of the present invention is now described with reference to FIGS. 12 - 15. Briefly, there are three types of controls which can be utilized to dynamically display run time values and state information according to the second embodiment of the present information. Namely, a needle gauge control, analog bar control, and point information block. A needle gauge control according to the present invention is shown in FIG. 12, an analog bar control is shown in FIG. 13, and a point information block control is shown in FIG. 14.

These controls are distinguishable from prior art building automation controls in several aspects. One of these aspects pertains to the modularity of the controls. Specifically, each of the controls of the present invention is a separate OCX stored in an individual data file. A user can elect which control points are monitored, and can specify the type of control used to monitor the control point. Further, each instance of a control is independent of other controls, with each control being refreshed, as needed, during its own processing cycle. In other words, the OCX controls of the present invention are independently and dynamically updated in a multi-threading environment.

More particularly, each control is an object composed of static and dynamic sub-objects. Static sub-object are those parts of the object which are not refreshed or repainted during a refresh cycle. By manner of illustration, in the needle gauge control 168 of FIG. 13, the scale 170, and control point name 172 are static sub-objects. In contrast, the arrow segment 174 and arrow head 176 are dynamic sub-objects because they are dynamically refreshed to reflect the current run time values of the control point. The independence of the static sub-objects from the dynamic sub-objects according to the present invention facilitates rapid multi-threading operations as processing resources during each processing cycle need only be utilized to refresh the dynamic sub-objects.

A method of creating a dynamic control will now be explained with reference to FIG. 15. This method closely parallels the method of creating an associated object

control, the main difference being the customizable options associated with each individual control.

The user selects a needle gauge, for example, from the group including needle gauge, analog bar, and point information block (block 182). Next the user selects a point to be linked or associated with the control (block 184).

Subsequently, the style characteristics for the selected control are specified (block 186). The ability of the user to modify the style and display options of each type of control is another important feature of the present invention. In contrast, prior art graphical controls in building automation systems were inflexible because the various style and display characteristics were coded into the graphical display program. Thus, modifying the size of a prior art control would require programming changes. In contrast, style and display characteristics of a control according to the present invention can be modified by a user without requiring programming changes.

For example, FIG. 5 shows an operator interface 82 for setting style characteristics for a needle gauge. Using the operator interface 82, the user elects whether to show an arrow head and/or the pivot point of the needle. Moreover, the user specifies the direction of rotation, pivot position, as well as the maximum and minimum needle deflection (in degrees).

In contrast, if the selected control was an analog bar, then the user would be able to select the orientation of the bar (block 186). For example, FIG. 16 shows an operator interface 200 wherein the user can choose a horizontally oriented analog bar or a vertically oriented bar. Further, the user can elect the direction in which bar will deflect. Thus, for a horizontally oriented bar, the user chooses whether the bar will increase to the left or to the right, and for a vertically oriented bar, the user chooses whether the bar will increase in an upward or downward direction.

Still further, if the type of control was a point information block, then in step 186, the user can select various style characteristics using an operator interface 210 (FIG. 17). Among other things, the user can elect whether to display the name of the associated control point, a descriptor, its status, and its alarm priority.

Then, in step 188, the user defines the state(s) for the control point, and specifies a unique color corresponding to each state. Thus, in addition to dynamically displaying run time values for a control point, state information is displayed using the colors associated with each state.

The user then selects display options for the selected control; such as a title to be displayed, whether to display the associated point name and alarm marks, as well as specifying the axis labels and associated fonts (block 190). Next, the user selects the state colors used for displaying each of the states which the control point can assume (block 192).

Finally, the state of the linked point is dynamically displayed by refreshing the dynamic-sub-objects in accordance with the specified alarm characteristics and run time values (block 194).

As discussed above, a user has the ability to specify the alarm state characteristics for each type of control. Thus, a user can specify that the color of the control change color to reflect the alarm state of the control point in addition to showing the actual run time value via needle deflection or displacement of the analog bar.

From the foregoing, it should be appreciated that a modular graphical display method for monitoring and controlling dynamic process information in a building automation system, which utilizes reusable control objects for graphically displaying real-time process state information has been shown and described. It should be further be appreciated that a building automation control system utilizing this modular graphical display method has been shown and described.

While various embodiments of the present invention have been shown and described, it should be understood that other modifications, substitutions and alternatives may be apparent to one of ordinary skill in the art. Such modifications, substitutions and alternatives can be made without departing from the spirit and scope of the invention, which should be determined from the appended claims.

Various features of the invention are set forth in the appended claims.

4. Brief Description of Drawings

DESCRIPTIONS OF THE DRAWINGS

FIGURE 1 illustrates a graphical application control object according to the present invention;

FIG. 2 illustrates a client with an interface to a needle gauge control object;

FIG. 3 shows a standard model used to access services provided by various kinds of software according to the present invention;

FIG. 4 shows an example of reusing an object through containment according to the present invention;

FIG. 5 shows a user interface for modifying properties of a control;

FIG. 6 illustrates a flow diagram for modifying the properties of a control;

FIG. 7 illustrates an Associated Point Control;

FIG. 8 illustrates a flow diagram of a method of associating a graphical object with a control point according to a first embodiment of the present invention;

FIG. 9 illustrates an operator interface for entering a point name to be associated with a graphic object;

FIG. 10 illustrates an operator interface for entering a state information for a control point name to be associated with a graphic object;

FIG. 11 illustrates a diagram showing data and command flows for creating a dynamic control overlaid on a static background picture;

FIG. 12 illustrates a Needle Gauge Control;

FIG. 13 illustrates an Analog Bar Control;

FIG. 14 illustrates a Point Information Block;

FIG. 15 illustrates a flow diagram for creating a dynamic control according to the second embodiment of the present invention;

FIG. 16 illustrates an operator interface for specifying style information for an analog bar control; and

FIG. 17 illustrates an operator interface for specifying a style information for an Information Block Control.

FIG. 1

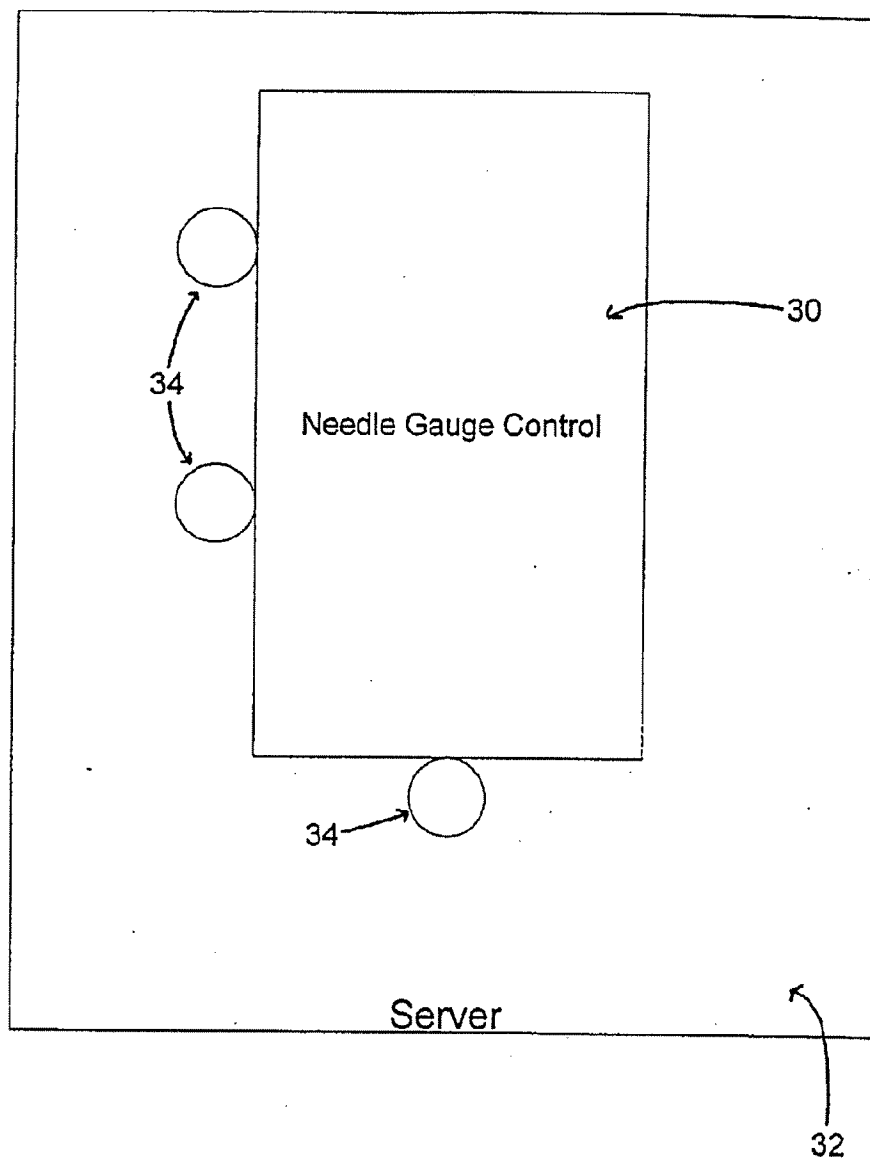


FIG. 2

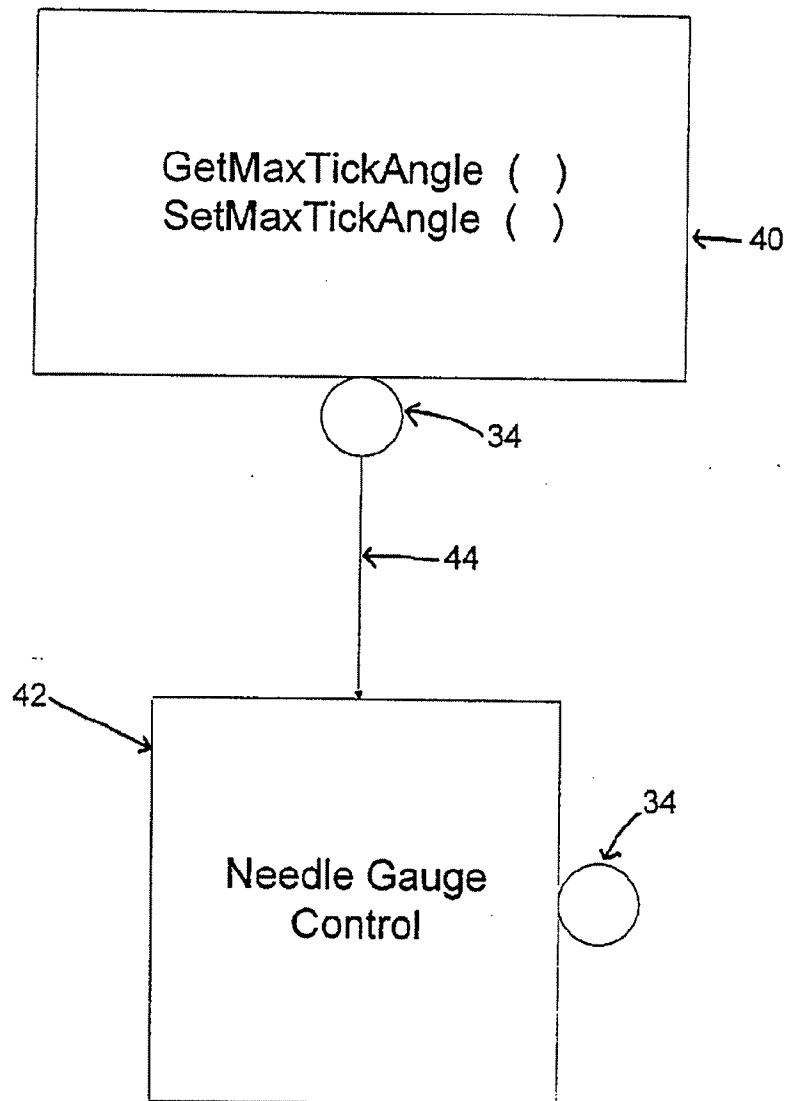


FIG. 3

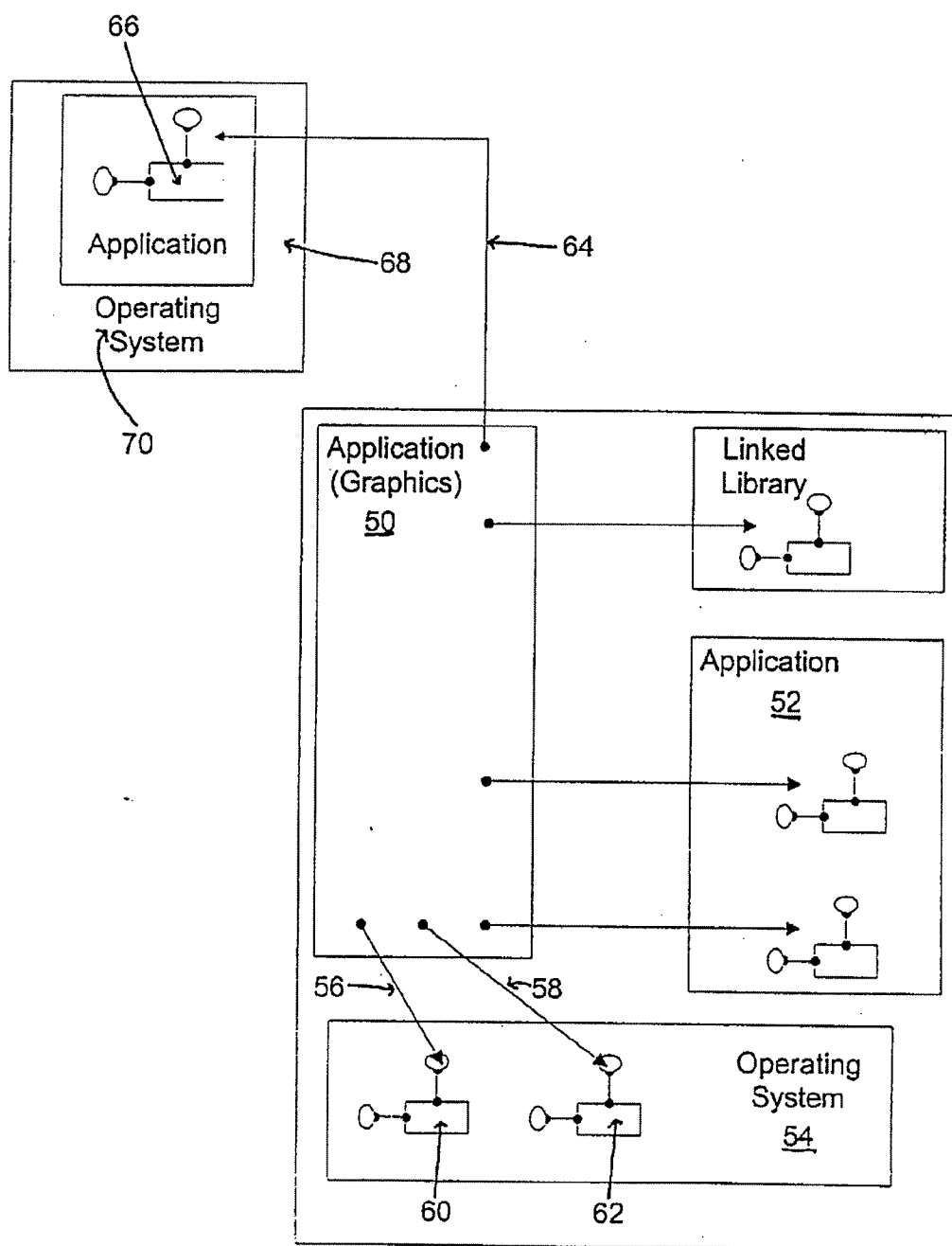
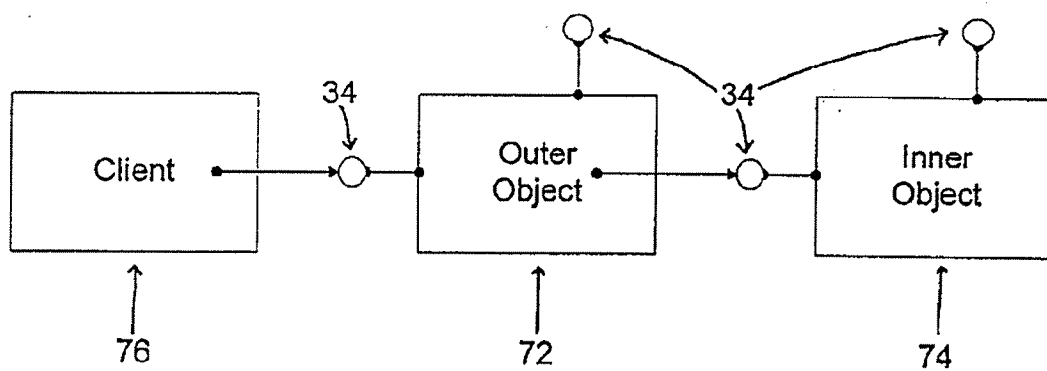


FIG. 4



Revising an object through containment.

FIG. 5

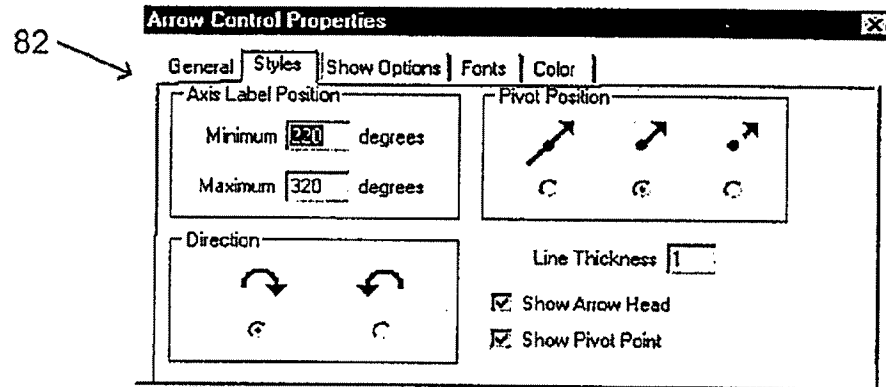


FIG. 6

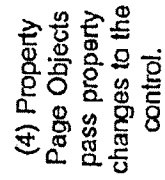


FIG. 7

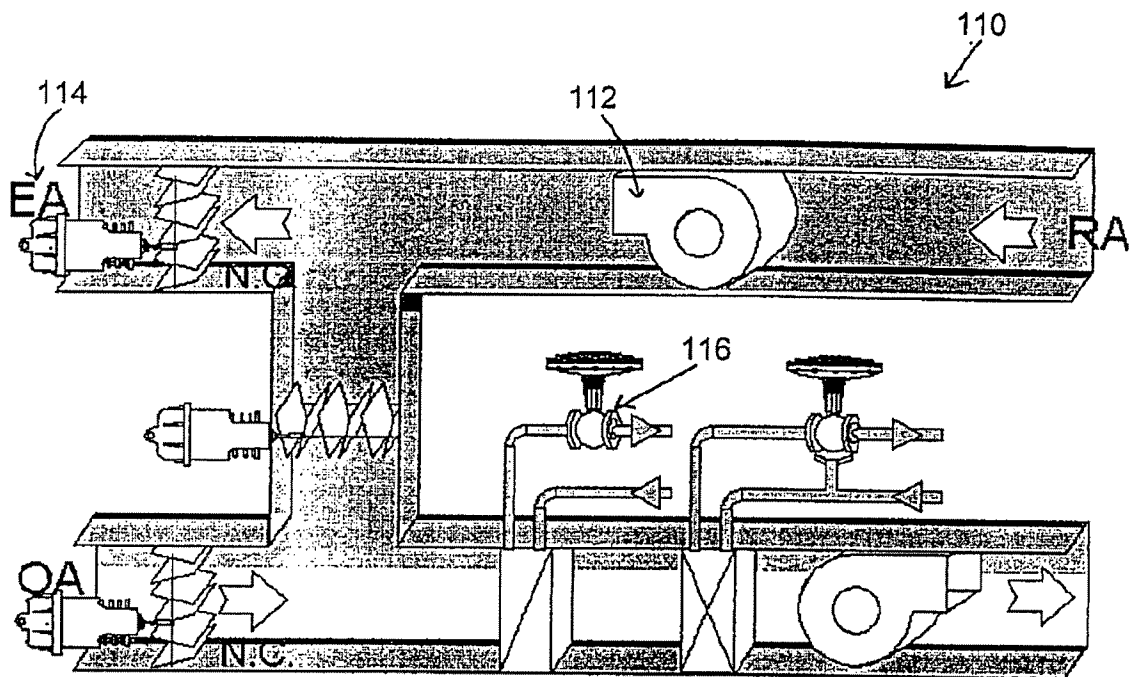


FIG. 8

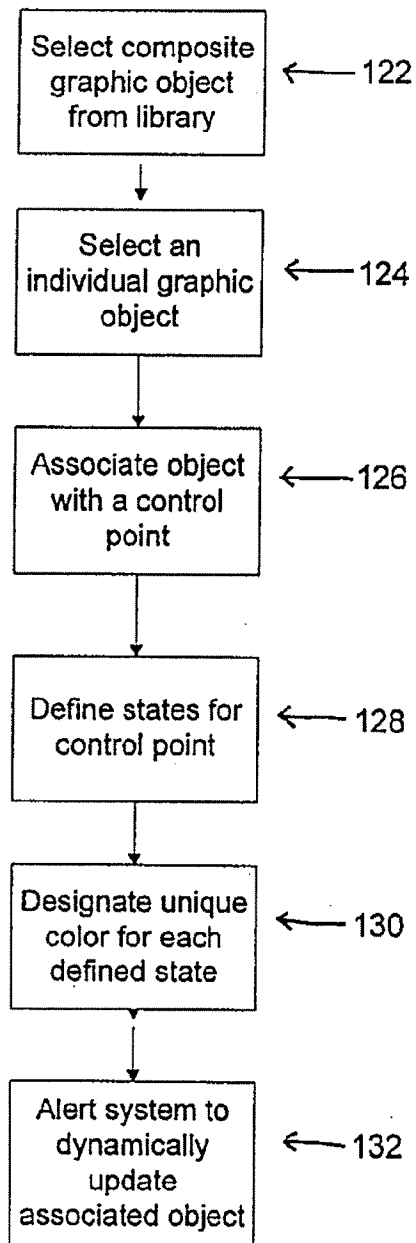



FIG. 9

Point Name: 

Configure Name To Display

Sample:

☒ Display Entire Name

☐ Display First Field of Name

☐ Display Last Field of Name


Field Delimiter: 

FIG. 10

Enhanced Alarm Setup

Alarm Destinations

001) Dest1

(None)

(None)

(None)

Mode Point:

LAI001

Mode Delay:

0

(min.)

Level Delay:

0

(sec.)

Differential:

0

☐ Acknowledge Return to Normal

Day

(Night)

(Special 2)

(Special 3)

(Special 4)

(Special 5)

☒ Alarm Mode Enabled

Set Point

Name:

Value:

50

	Offset	Priority	Extra Destination	Enhanced Alarm Message
1)	10	PRI1	(None)	(None)
2)	20	PRI2	(None)	(None)
3)	30	PRI3	(None)	(None)
4)	40	PRI4	(None)	(None)

FIG. 11

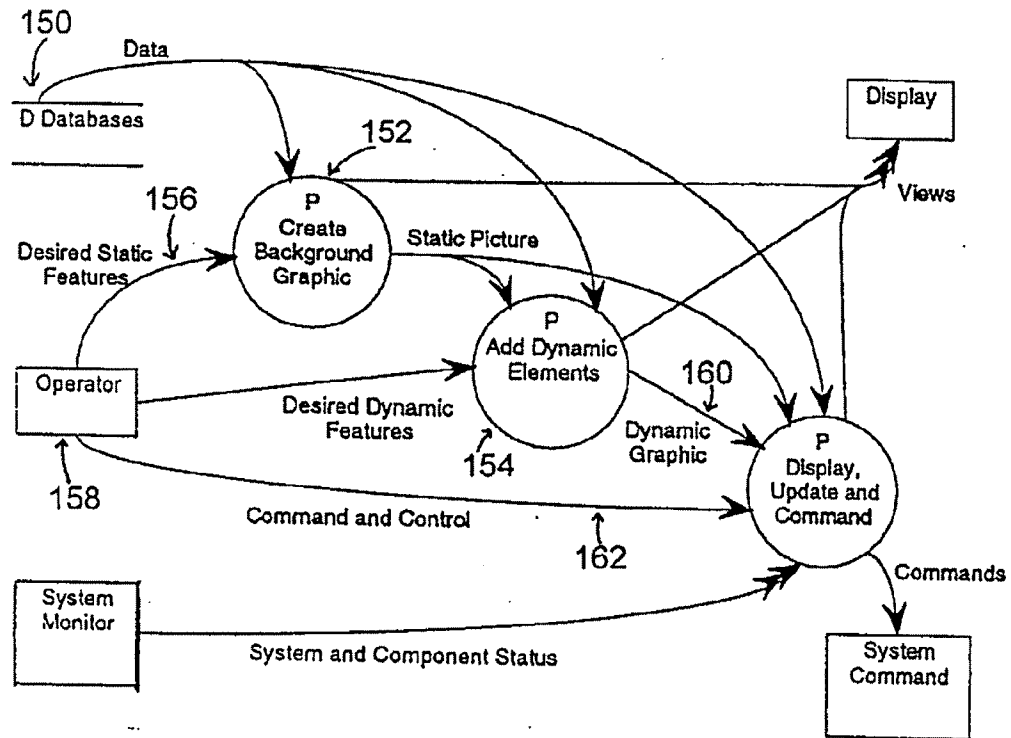


FIG. 12

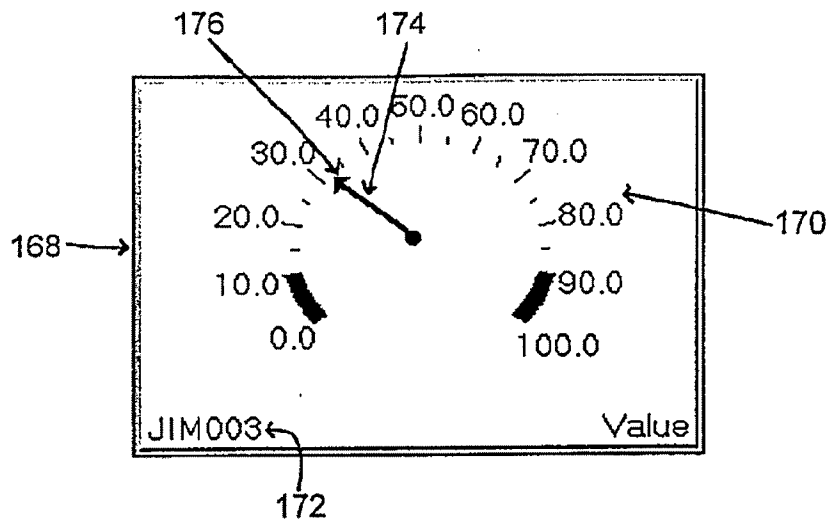


FIG. 13

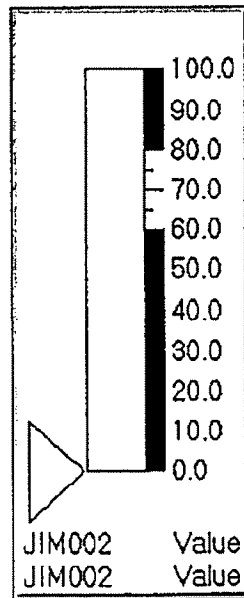


FIG. 14

JIM002U	JIM003
JAMES POINT	TEST POINT
Status	Status
Priority	Priority
Value	Value

FIG. 15

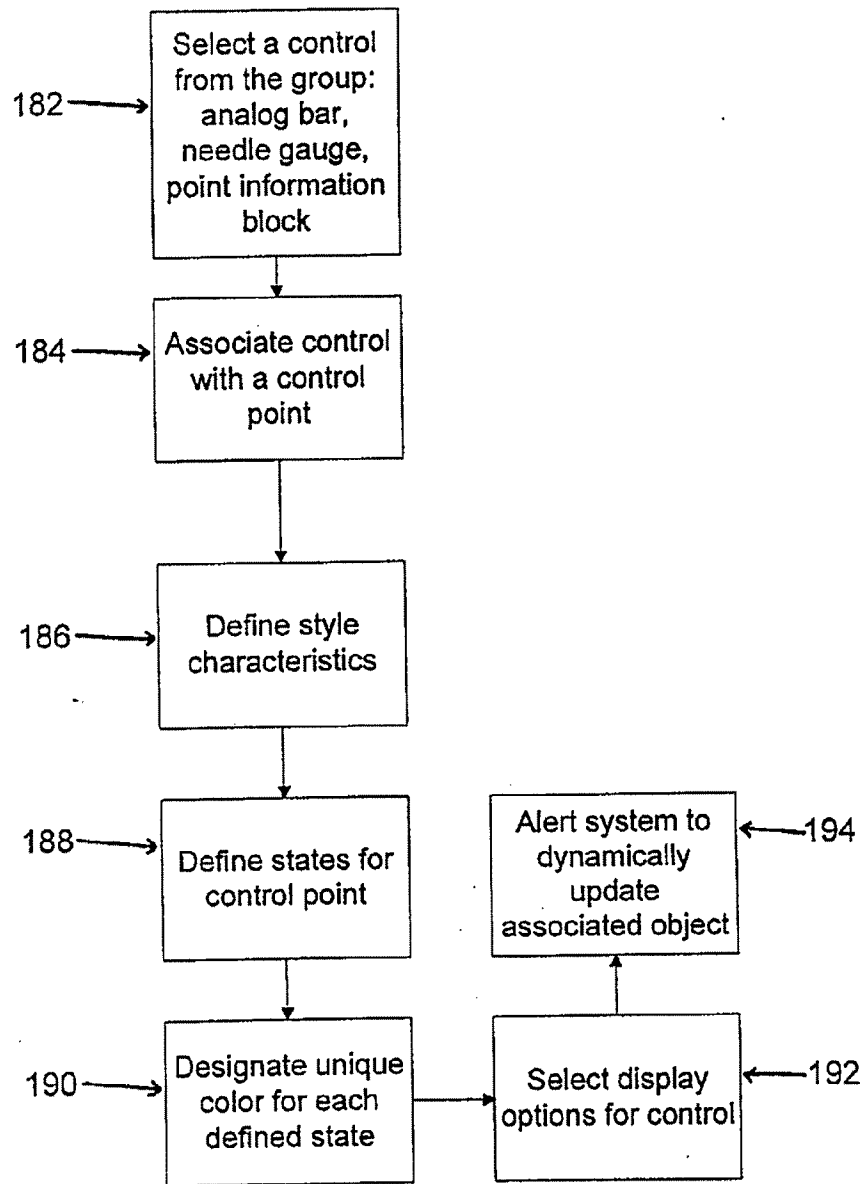


FIG. 16

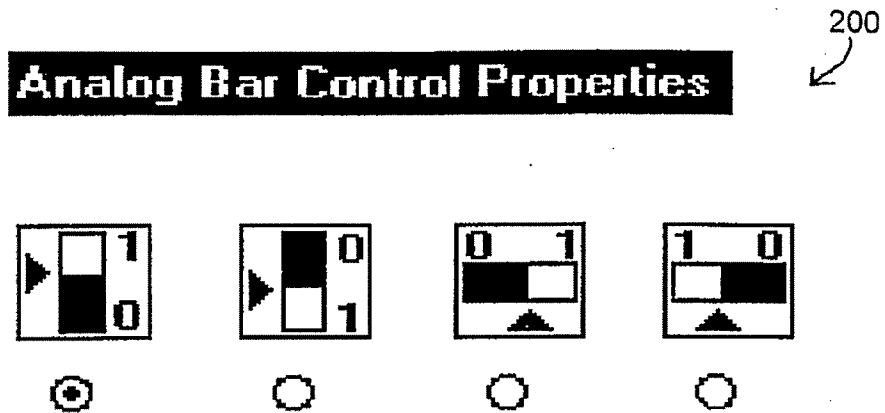


FIG. 17

210

Fields

- ☒ Display Name
- ☒ Descriptor
- ☒ Status
- ☒ Priority
- ☒ Current Value ☒ Units
- ☐ Totalized Value

Display Style

- ☒ Multiple Lines ☐ Single Line

State Color

- ☒ Use For Background Rectangle
- ☐ Use For Font Color
- ☐ Use Reverse Video Instead

☒ Border

1. Abstract

The building automation control system of the present invention includes a supervisory control station for monitoring and controlling process information (control points) from at least one environmental control device. The supervisory control station is provided with a memory for storing data, including a dynamic link library (DLL) containing: first data for displaying a plurality of independently selectable graphic objects, and link data specifying an association between a selected graphic object and the control point. The supervisory control station dynamically displays the received real-time process state information for the associated control point using stored state table data, link data, and run time values from the environmental control device.

2. Representative Drawing

Fig. 8